Smart Agricultural Technology 12 (2025) 101324

Contents lists available at ScienceDirect

Smart Agricultural Technology

journal homepage: www.journals.elsevier.com/smart-agricultural-technology

Incremental learning with domain adaption for tomato plant phenotyping

Angelo Cardellicchio [a,1], Vito Renò [a,*,1], Francesco Cellini [b], Stephan Summerer [b], Angelo Petrozza [b], Annalisa Milella [a]

[a] Institute of Intelligent Industrial Technologies and Systems for Advanced Manufacturing, National Research Council of Italy, Via Amendola 122, D/O, Bari, 70126, Italy
[b] ALSIA Centro Ricerche Metapontum Agrobios, s.s. Jonica 106, km 448.2, Metaponto, 75010, Italy

ARTICLE INFO

MSC:
0000
1111

Keywords:
Plant phenotyping
Incremental transfer learning
Trait detection
YOLO

ABSTRACT

A comprehensive and continuous analysis of relevant phenotypic traits in broadly diffused crops such as tomatoes is extremely important to assess the status of plants, especially in the current scenario of extreme climate events, which can threaten the foundations of the food supply chain for large populations. However, these operations are extremely costly in human effort; therefore, automating these processes is of paramount relevance. To this end, this work first provides a systematic benchmark of different, well-assessed versions of the You Only Look Once (YOLO) object detector, identifying the most suited baseline in terms of iteration and model density. Afterwards, the effectiveness of attention-based mechanisms was evaluated, highlighting possible critical aspects which may undermine overall results and real-time applicability. Thus, to enhance the baseline performance, an Incremental Learning pipeline was assessed, evaluating domain adaptation via fine-tuning. Specifically, a YOLOv11-based object detector was incrementally trained on specific subsets of a larger dataset, with the aim of providing the resulting model with subset-specific knowledge to allow more generalisation capabilities. This procedure resulted in an overall improvement of mAP@0.5 of about 1.36%, and F1 score of 1.1%, while slightly lowering the computational burden during inference, with an improvement of 19 ms. The effectiveness of these capabilities was also tested against a scenario with a low number of samples available, yielding promising domain adaptation results even under adverse conditions, and providing a practical path for evolving phenotypical evaluation under real-world agricultural scenarios.

1. Introduction

Tomatoes are among the most consumed fruits in the world [12]. This crop is extremely popular in the United States, China, and Italy, accounting for about 65% of the overall global consumption due to tradition and cultural motivations. Still, tomato production is expected to shrink over the next few decades due to the negative impacts of climate change, with a projected decrease of 66% in the overall production compared to the levels of 1989 [2]. Consequently, assessing and evaluating these impacts is extremely valuable in providing an economically viable yet sustainable yield for this type of crop.

One of the main investigation methods researchers have used over the last few decades to assess the status of a plant subject to the stress of genotypical modification is phenotyping. Specifically, this term describes a set of non-destructive tests, usually performed visually, which can help the domain expert in assessing the status of a plant via its phenotypical traits. To gather statistical significance and properly characterise a data generation mechanism for certain aspects of phenotypic traits, gathering a significant amount of data is mandatory. To deal with these issues, high throughput phenotyping (HTP) platforms were largely developed and deployed over recent years [15], providing high quantities of data which practitioners and researchers could use to develop a generalizable and effective tool for automating part of the evaluation of large cultivars and fields. Still, due to the amount of data generated by these platforms, an evaluation fully performed by domain experts is not viable. Consequently, the development of automatic systems supporting human operators is mandatory.

To this end, one of the most interesting applications is deep learning-based object detection, which can be used to localise and classify objects of interest within the scene. These types of networks can be broadly cate-

* Corresponding author.
E-mail address: vito.reno@stiima.cnr.it (V. Renò).
[1] These authors equally contributed to the work.

https://doi.org/10.1016/j.atech.2025.101324
Received 26 May 2025; Received in revised form 31 July 2025; Accepted 13 August 2025

Available online 21 August 2025

gorised as follows. First, there were two-stage detectors, which are based on two cooperating networks mainly based on convolutional layers. Specifically, the first network is often called a region proposal network, and its role is the identification of possible regions of interest within the image. As for the second network, its role is to identify the most relevant class for each proposed region of interest. These networks are usually highly accurate, but have significant inference costs due to the presence of two different networks. A significant reduction of the computational burden was achieved by single-stage detectors, which use a single network to perform localisation and classification in a single step. Among these classifiers, the most well-known are the ones belonging to the You Only Look Once (YOLO) family, which is currently being developed with numerous iterations. Newer approaches are called end-to-end detectors, and their aim is to fully automate the detection pipeline, removing manually set thresholds or the identification of an initial set of anchors, at a higher computational cost due to the presence of computationally intensive self-attention mechanisms. Therefore, detectors derived from YOLO are the most frequently used in the domain of plant phenotyping, mainly due to their balance in terms of speed and accuracy. Specifically, researchers have focused on improving the base architecture to deal with problems specific to plant phenotyping, such as the requirement to identify small phenotypical traits (e.g., flowers or fruit) which may not be gathered under optimal conditions. To deal with this issue, several works focused on adding attention mechanisms within the base YOLO network. For example, researchers in Chen et al. [4] integrated attention within the detection head, while researchers in Gao et al. [6] focused on the backbone of the base YOLOv5 model. In both cases, this targeted improvement allowed to achieve enhanced performance if compared to the baseline. Other works, such as Solimani et al. [14] and Sun [16], focused on integrating Squeeze-and-Excitation attention mechanisms in YOLOv8, also in this case improving the overall performance, especially when dealing with visually small phenotypical traits. Another issue is related to the deployment of object detectors on constrained hardware mountable on agricultural robots. This specific issue undermines the applicability of detectors based on transformers, which were proposed over the last few years and confirmed superior performance at the cost of a higher computational load, even if lightweight approaches were proposed, for example, in Wang et al. [18]. In other cases, the authors focused on improving well-assessed YOLO-derived detectors, for example, by using lightweight networks such as MobileNetV3 in the backbone of the model [23], or by reducing the computational burden of the feature pyramid used for feature fusion in the neck of the network [25,21].

While effective, current approaches often overlook the underlying assumption that data belonging to phenotypical scenarios is static, that is, coming from the same data generation mechanism. This may be challenging to demonstrate in complex, outdoor scenarios, where imagery may be gathered at different time points, representing different instants in the growth of the plant. Furthermore, the generalisation capability of a model should be tested not only at different growing stages but also when different settings and conditions are considered. To address these issues, researchers usually focus on gathering large, comprehensive datasets, collecting data over time to represent different fields, growth stages, or even sensors. However, training a model from scratch on these large datasets is computationally intensive and does not address the intrinsic non-stationarity in the data, as new information can be gathered and added at any given time.

These considerations highlight a gap for current research, that is, the requirement for an incremental update of existing models using only new data batches. To this end, this work proposes and evaluates an incremental transfer learning framework to solve this sequential adaptation problem. Furthermore, the impact of the integration of two attention mechanisms, that is, Convolutional Block Attention (CBAM) and Shuffle Attention (SA), is evaluated.

Consequently, the main contributions of this work are the following.

- First, a direct comparison between different versions of YOLO, that is, YOLOv5, YOLOv8, and YOLOv11, is proposed to identify the optimal model for the task at hand.
- Then, a novel Incremental Learning framework was proposed, demonstrating its effectiveness and significant improvements of both accuracy and training stability over standard methods.
- A critical analysis of CBAM and SA is provided, revealing that while these mechanisms are effective in the case of smaller models, they may not be relevant in the case of denser architectures.
- Finally, the practical applicability of the proposed Incremental Learning Framework is discussed, adapting the trained model to a loosely coupled scenario via domain adaptation and fine-tuning.

The rest of the paper is organised as follows: Section 2 reports the materials and methods of the work, with focus on the dataset used, the reference architecture for the experiments, the attention modules implemented, as well as the learning approaches adopted and the evaluation metrics. Section 3 recaps the experimental results. Section 4 discusses the achieved results, highlighting the impact of Incremental Learning with the help of explainability methods. Finally, Section 5 draws the conclusions and future works.

2. Materials and methods

2.1. Dataset description

The datasets used in this work were originally proposed in Cardellicchio et al. [3] and will be referred to as *in-lab* and *in-field* in the following.

2.1.1. In-lab dataset

The *in-lab* dataset was gathered using the LemnaTec Scanalyzer3D system located at the HTP platform of the PhenoLab of the ALSIA Metapontum Research Center. The idea was to automatically convey tomato plants to imaging chambers, gathering three views for each plant, one top view and two 90 degrees displaced side-views. Overall, 1673 RGB images were gathered, each with a resolution of 1624×1234 pixels. Afterwards, domain experts used the CVAT labelling tool to identify three relevant phenotypical traits in the dataset – namely flowers, fruit, and nodes – that were annotated using rectangular bounding boxes. Fig. 1 shows a sample of the labelled dataset.

2.1.2. In-field dataset

The *in-field* dataset was collected at an experimental field set up on a commercial farm in Jolanda di Savoia, next to Ferrara, in Italy. The field included three tomato varieties, i.e., San Marzano, Mariner F1, and UC 82, organised according to a randomised block setup. Data were acquired at the ripening time, just before harvesting, in September 2022, over a two-day campaign performed with the equipment shown in Fig. 2. After the experimental campaign, tomato berries were manually collected and counted to provide ground truth. Specifically, the plants located within a 1.5 m transect for each row were considered for manual counting. Colour images with a 1280×720 resolution were acquired using an Intel RealSense D435 camera at different daylight conditions. The camera was positioned with the optical axis orthogonal to the field plane. Among the overall dataset, 26 images were acquired in the areas where manual counting was performed and were used as an independent test set. Some sample images from the three varieties are shown in Fig. 3. Manual labelling was performed by domain experts to annotate ripe tomatoes, labelled as red berries (*rb*), and unripe tomatoes, labelled as other berries (*ob*). This kind of classification would be useful in automated harvesting applications.

2.1.3. Datasets comparison

Even if both the *in-lab* and *in-field* datasets hold images of tomato plants, they represent two distinct domains, making them suitable for

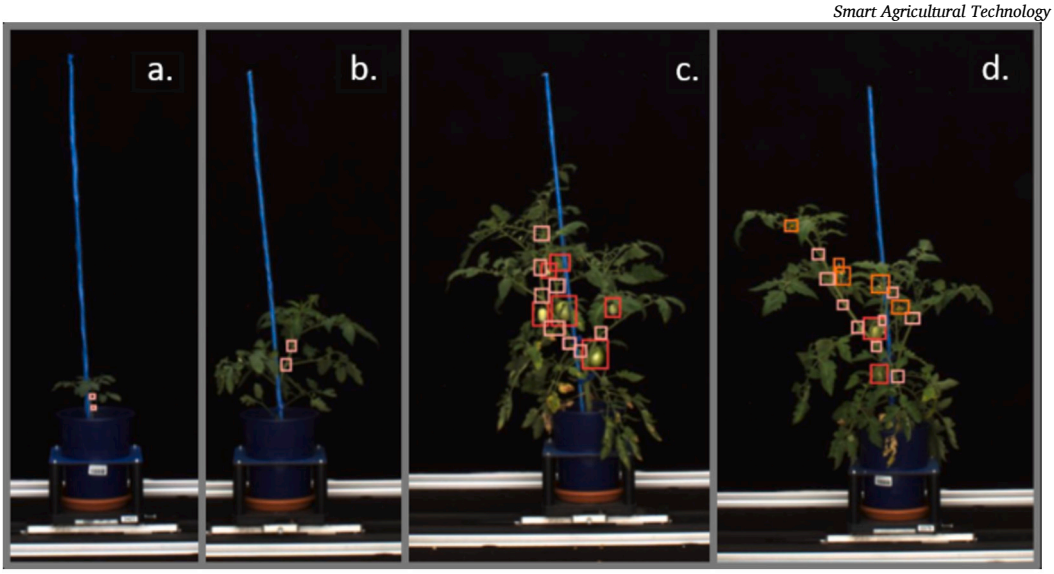

Fig. 1. A sample of the images collected in the lab with the PhenoLab HTP platform.

Fig. 2. The setup for the field experiment.

Fig. 3. Sample images collected in the field for each variety: (a) Mariner F1; (b) San Marzano; (c) UC 82.

cross-domain generalisation analysis. The key differences, summarised in Table 1, are the following.

- **Imaging setup:** On the one hand, the *in-lab* dataset was collected within an HTP platform, which inherently guarantees consistent acquisition conditions throughout different experiments in terms of lighting, pose, occlusions, and background. On the other hand, the *in-field* dataset was gathered under variable, daylight conditions, thus including shadowing effects, occlusions, and complex background.
- **Sensing technology:** The *in-lab* dataset was captured with a LemnaTec Scanalyzer 3D, which provided high-quality images with a resolution of 1624×1234. The *in-field* dataset, instead, used a much more limited Intel RealSense D435, which provided lower resolution images (1280×720), along with differences in colourimetry and lens characteristics. Therefore, the *in-lab* dataset represented an ideal condition using a high-end sensor, while the *in-field* dataset represented a real-world scenario where a consumer-grade sensor was used.
- **Scope and semantics**: The *in-lab* dataset includes three classes, that is, flowers, fruits, and nodes, aimed at plant phenotyping. The *in-field* dataset has two classes (ripe and unripe berries), aimed at a specific harvesting application.

Table 1
Comparison of In-Lab and In-Field Datasets.

Characteristic	In-Lab Dataset	In-Field Dataset
Lighting variability	None	High
Background	Uniform	Complex
Sensor	LemnaTec Scanalyzer3D	Intel RealSense D435
Resolution	1624×1234	1280×720
Viewpoint	Side	Top-down
Occlusion	Low	High
Classes	3 (flower, fruit, node)	2 (ripe, unripe)
Scope	Phenotyping	Harvesting automation

2.2. YOLOv11 architecture

The models used in this work belong to the *You Only Look Once* (YOLO) family of single-stage detectors, originally proposed by Redmon in Redmon et al. [13]. YOLO is a single-stage object detector aiming to simultaneously locate and classify objects of interest within images. YOLO detectors share a common architecture composed of the following building blocks:

- *Backbone*: the backbone is the part of the network delegated to identify relevant features in the image.
- *Neck*: the neck gathers the features extracted by the backbone and fuses them at different levels to embed local and global contextual information.
- *Head*: the detection head performs the final decision on the localisation and classification of objects within the image.

The YOLO family evolved through several iterations, reaching its 11^{th} version in 2024 [9]. This iteration was a direct evolution of YOLOv8 [8], which primarily used in its architecture a modified version of the Cross Partial Stage (CSP) [17] layer named *CSP bottleneck with two convolutions* (C2f). Specifically, YOLOv11 introduced a newer, more efficient version of the C2f layer named C3k2, whose main novelty was replacing large convolutions in the C2f block with two smaller convolutions (i.e., a smaller kernel size). Furthermore, the YOLOv11 architecture introduced the *Cross Stage Partial with Spatial Attention* (C2PSA) block in the neck, aiming to focus on the most relevant regions of the image. Finally, in the head, YOLOv11 introduced the *Convolution-BatchNorm-SiLU* (CBS) block, further refining feature maps used for detection, normalising the data flow through batch normalisation, and using a sigmoid linear unit activation function as the activation function to introduce non-linearities. The standard architecture of YOLOv11 is shown in Fig. 4.

In this work, YOLOv11 was selected as the primary architecture for two different reasons. First, it embeds recent components that are well-suited for challenges related to complex scenarios, such as plant phenotyping. Specifically, the C3k2 module is designed for finer-grained feature extraction, which improves the localisation of small phenotypical traits within the *in-lab* and *in-field* datasets. Second, YOLOv11 demonstrated a superior accuracy trade-off when compared to older iterations, such as YOLOv5 and YOLOv8 [1], including in agricultural applications. These factors allowed the selection of YOLOv11 as a robust baseline architecture for the evaluation.

Thus, even if an attention mechanism is already embedded within the C2PSA block, it is worth pointing out that the objective of this work was to evaluate whether the use of attention mechanisms within the neck could further improve the detection performance of small, localized objects, which are of great relevance in the context of the identification of phenotypic traits, especially in tomato crops images. In particular, the modifications to the standard YOLOv11 architecture aimed to assess whether a second attention mechanism, focused on both channel and spatial attention, could help the neck to improve the detection capabilities of the whole network, thanks to the feature fusion approach implemented between the backbone and the neck. Consequently, the standard architecture was modified as shown in Fig. 5. Let us now discuss the details of the attention modules chosen for the analysis.

2.3. Attention modules

2.3.1. Motivation

One of the most assessed methods to improve the ability of models based on convolutional layers to focus on relevant information within the image is to use attention mechanisms. Broadly speaking, two main categories of attention exist: *spatial* attention, which helps the network in identifying the relevant spatial part of the feature map in defining the decision, and the *channel* attention, that is, attention mechanisms which allow the network to focus on specific features within the feature map.

Attention mechanisms were successfully integrated in YOLO-based detectors, demonstrating superior performance, for example, in Chien et al. [5], where the insertion of the CBAM [20] was shown to outperform several other attention mechanisms for pediatric wrist fracture detection. Another interesting improvement was provided by the SA mechanism, originally proposed in Zhang and Yang [24], demonstrating improved performance for fine-grained defect detection in the textile context if compared to the baseline. These mechanisms were also successfully tested and adopted in the agricultural context, for example, for pomegranate detection [26].

This work evaluates the impact of the integration of CBAM and SA attention within the YOLOv11 neck. Apart from the promising results achieved by other works, this was also motivated by the contextual use of spatial and channel attention proposed by these two methods. In other words, combining channel and spatial attention can improve the capabilities of the network to localise areas of the feature map where the phenotypical traits are available, therefore improving small object detection, while improving in discerning between similar and potentially ambiguous traits and levels of ripeness of the berries.

2.3.2. Convolutional block attention module

The CBAM attention mechanism was first proposed in Woo et al. [20] to model both channel and spatial attention. Specifically, given an intermediate feature map denoted as $F_i \in \mathbb{R}^{C \times H \times W}$, with C number of filters, and H and W height and width of the attention map, respectively, the CBAM block provides a *channel refined* feature F_{CR} starting from the inferred single channel attention map $M_C \in \mathbb{R}^{C \times 1 \times 1}$.

$$F_{CR} = M_C(F_i) \otimes F_i \tag{1}$$

In Equation (1) and in the following, the symbol $\otimes$ represents the element-wise multiplication. Let us underline how $M_C(F)$ is computed as the element-wise summation of the descriptors extracted by a Global Average Pooling (GAP) and a Global Max Pooling (GMP) to aggregate the information via a multilayer perception MLP, provided as an input to a sigmoid function.

$$M_C(F) = \sigma\left[MLP(GAP(F)) \oplus MLP(GMP(F))\right] \tag{2}$$

In Equation (2), the $\oplus$ symbol represents the element-wise summation. Once F_{CR} is known, the final refined feature F_{FR} can be computed as follows:

$$F_{FR} = M_S(F_{CR}) \otimes F_{CR} \tag{3}$$

As for M_S, it is computed using a convolutional layer with filter size 7×7, as described in Equation (4).

$$M_S(F) = \sigma\left[f^{7\times7}(GAP(F) \circ GMP(F))\right] \tag{4}$$

The overall output feature map of the CBAM block is then given by the element-wise summation of the input feature maps and their final refined version.

$$F_o = F_i \oplus F_{FR} \tag{5}$$

The structure of the CBAM module is shown in Fig. 6.

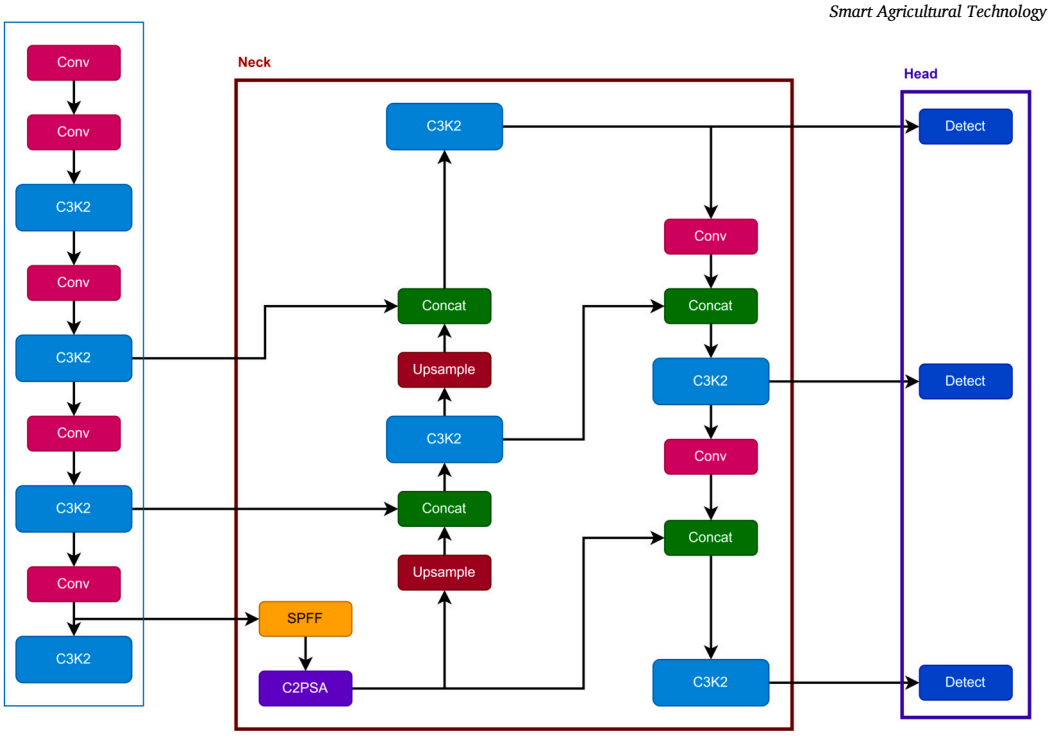

Fig. 4. The standard YOLOv11 [9] architecture.

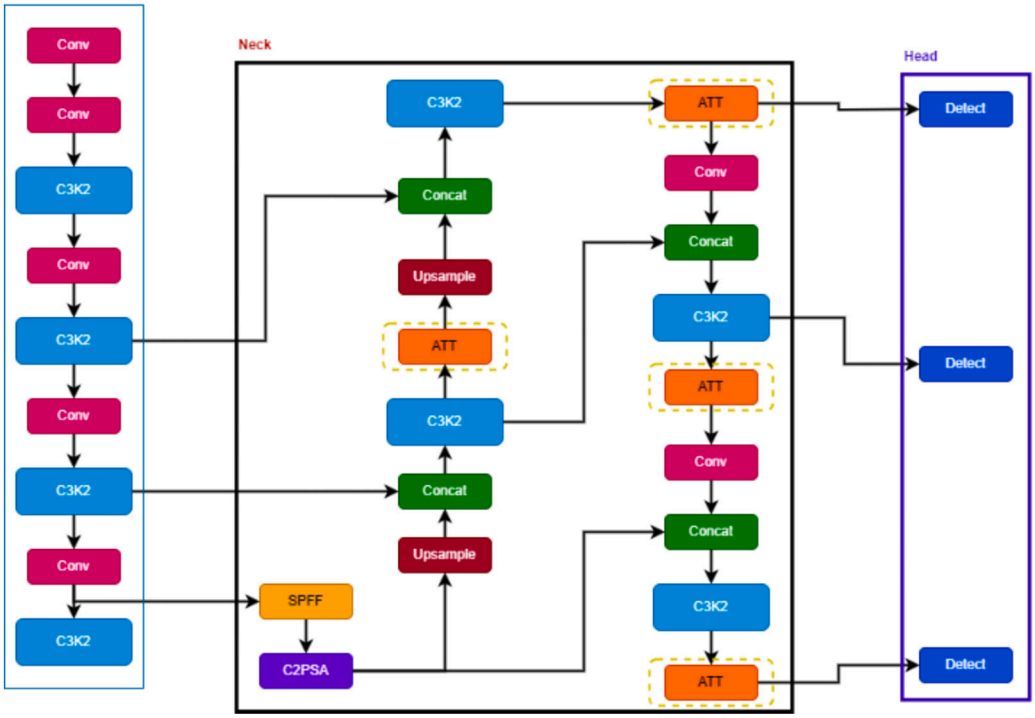

Fig. 5. The modified YOLOv11 [9] architecture. The attention modules (SA [24] or CBAM [20]), highlighted within a yellow rectangle, were inserted after the C3k2 blocks to enhance feature selection using channel and spatial attention.

2.3.3. *Shuffle attention*

The SA module was first introduced in Zhang and Yang [24] and proposes another way to combine channel and spatial attention through feature grouping and a shuffle mechanism, as shown in Fig. 7.

Specifically, the SA module starts from the assumption that an input feature map is composed of G sub-features x_i, with $i \in 1, \ldots, G$, each one further splittable into two sub-features, namely x_{i_1} and x_{i_2}. Consequently, x_{i_1} is used to compute the channel attention $x_{i_1 C}$ as follows:

$$x_{i_1 C} = \sigma\left[FC\left(GAP(x_{k_1})\right)\right] \otimes x_{k_1} \tag{6}$$

In Equation (6), FC represents a fully connected layer. As for the spatial attention $x_{i_2 S}$ it is computed as follows.

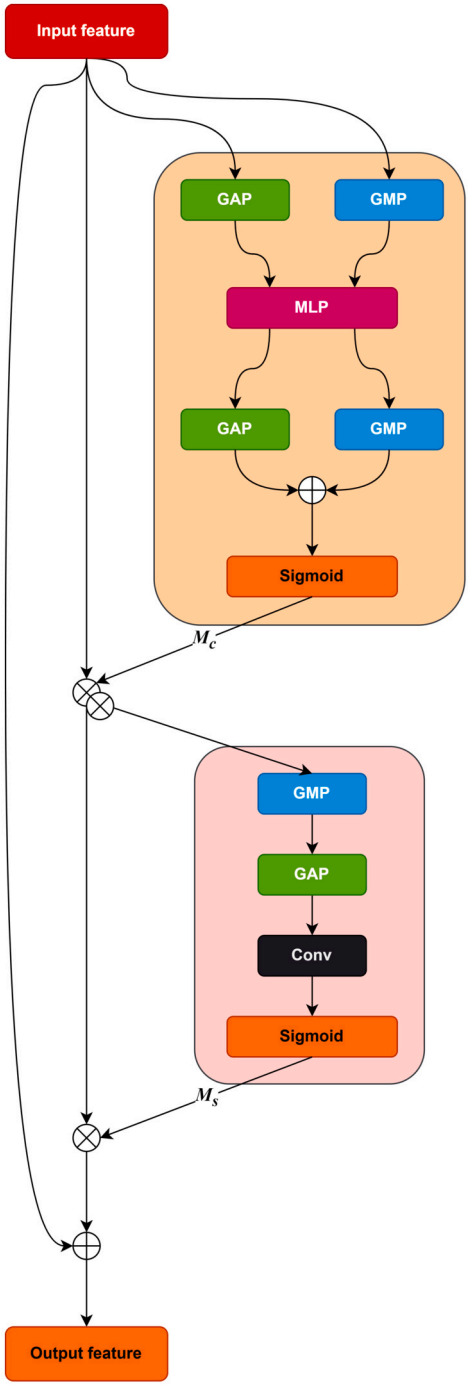

Fig. 6. The CBAM [20] block scheme.

$$x_{i_2S} = \sigma\left[FC\left(GN(x_{k_2})\right)\right] \otimes x_{k_2} \tag{7}$$

In Equation (7), GN refers to a group normalisation layer. The SA module concatenates the channel and spatial attention, obtaining an overall sub-feature x'_i, which is finally aggregated using a channel shuffle operator to get the final output feature.

2.4. Evaluation metrics

The results were evaluated based on several metrics commonly used in object detection, that is, *precision* (P), *recall* (R), $F1score$ ($F1$), and *mean Average Precision* (mAP). These metrics can be evaluated starting from the quantities defined afterwards.

2.4.1. Positives and negatives

For the sake of simplicity, let us briefly consider a problem where only two types of objects should be detected. In this case, the problem can be framed as binary classification, thus a *positive* and a *negative* class can be defined. Consequently, the following quantities can be specified.

- **TP** represents the *true positives*, the instances correctly identified by the model as belonging to the positive class.
- **TN** represents the *true negatives*, the instances correctly identified by the model as belonging to the negative class.
- **FP** represents the *false positives*, that is, the instances of the negative class incorrectly identified by the model as belonging to the positive class.
- **FN** represents the *false negatives*, that is, the instances of the positive class incorrectly identified by the model as belonging to the negative class.

It is important to underline how these metrics should be enriched when evaluating object detection problems, such as in the YOLO case, by considering the so-called background quantities:

- **B-FP** represents the *background false positives*, i.e., the instances identified by the model on the background. In this case, the model output could be a correct instance (not labelled in the original dataset) or an incorrect one.
- **B-FN** represents the *background false negatives*, i.e., the instances labelled in the original dataset but not identified by the model.

2.4.2. Precision, recall, and F1 score

Given these quantities, precision P and recall R can be computed as follows.

$$P = \frac{TP}{TP + FP} \tag{8}$$

$$R = \frac{TP}{TP + FN} \tag{9}$$

The formulation for the $F1$ score can be directly derived from Equations (8) and (9).

$$F1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{10}$$

2.4.3. Mean average precision

The evaluation of the mAP requires the introduction of the concept of *Intersection over Union* (IoU), which is defined as:

$$IoU = \frac{A_O}{A_U} \tag{11}$$

In Equation (11), A_O is the overlap area between the ground truth from the labelling and the corresponding predicted box, while A_U is the union between these areas. The value for the IoU is defined within the range $[0, 1]$ and is directly proportional to the overlap between the predicted box and its corresponding ground truth. In practice, when $IoU = 1$, the predicted box completely overlaps the ground truth, while when $IoU = 0$, no ground truth pixels are overlapped with the predicted bounding box.

Given a certain IoU threshold, one can compute the *average precision* (AP) as the area under the *precision-recall* curve for that threshold. As the AP value is computed for each class, the mAP is the average value of the AP over all classes. In the experiments, the mAP@0.5 was evaluated. This was motivated mainly due to the small size of the phenotypical traits, for which an overlap and a confidence above 50% may be considered acceptable.

2.4.4. Computational complexity

Finally, the computational complexity of the model is assessed via the inference time on a small-sized NUC used for the trial to simulate real-world settings.

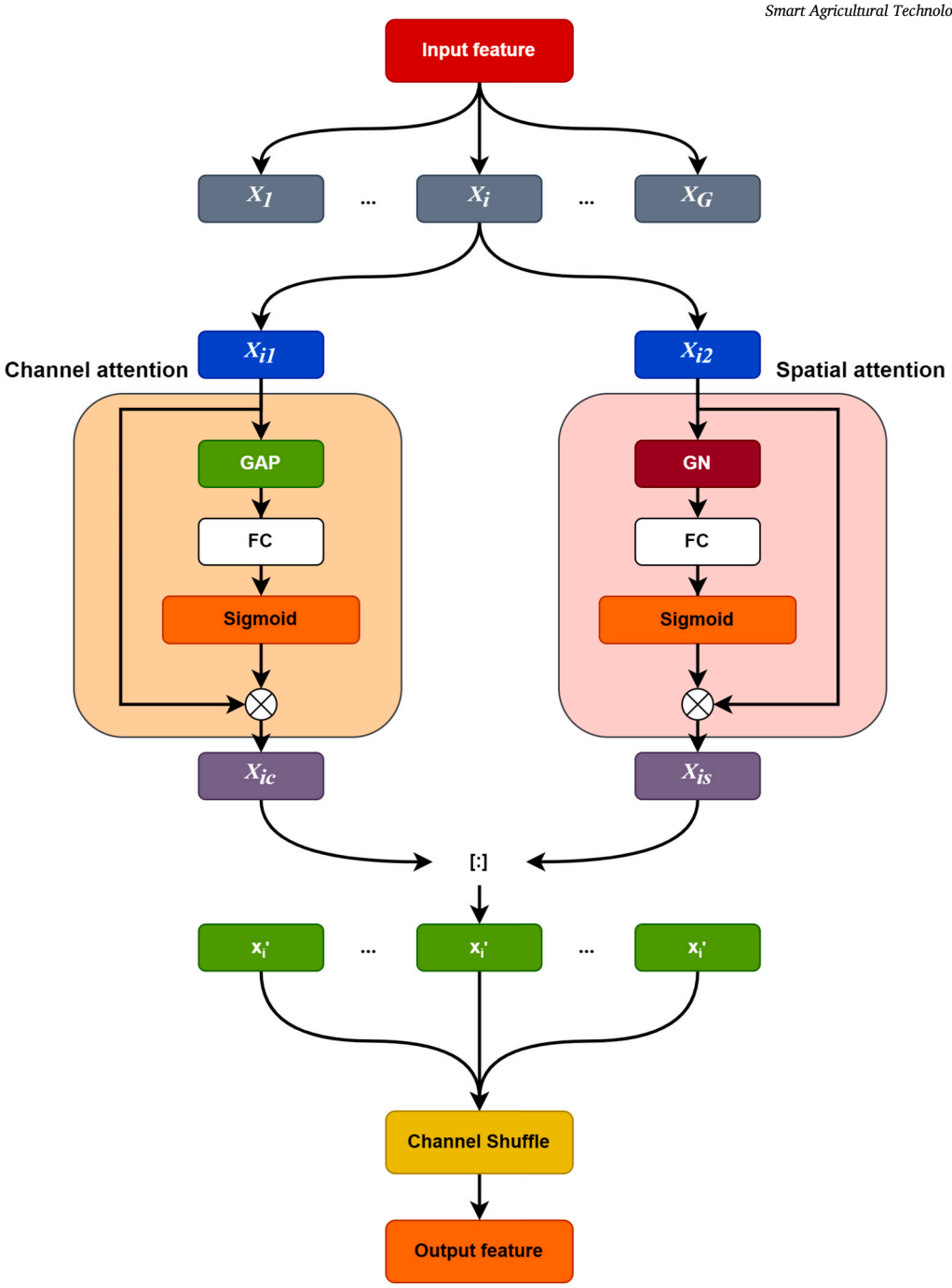

Fig. 7. The SA [24] block scheme.

2.5. Incremental transfer learning

Real-world agricultural scenarios, ranging from in-field deployments to controlled settings in HTP platforms, often imply several temporally distributed data-gathering campaigns, which usually span different growth stages of the plants. This implies that the datasets are increasing over time to reflect novel phenotypical conditions. However, it is often computationally expensive to continuously retrain algorithms on the entire incremental datasets each time new data becomes available. Therefore, this work proposes an incremental transfer learning strategy, designed to allow sequential fine-tuning specifically in these situations, with the aim of updating existing models with novel information without the requirement for full retraining on all the past data.

To this end, the *in-lab* dataset, which contained samples of plants at different growth stages, was partitioned into κ disjoint, sequential subsets $ds_1, ds_2, \ldots, ds_\kappa$, of equal sizes, each one representing a contribution from a different acquisition. The training process is then performed incrementally, starting from the initial subset ds_1, and being repeated across all the κ subsets. The iterative update can be described by the Equation (12).

$$\omega_{i+1} = \Phi(ds_{i+1}, \omega_i) \tag{12}$$

Here, ω_i represents the weights for the model obtained after training on subset ds_i, while Φ represents the fine-tuning process, which extends the knowledge and the generalisation capabilities of the model by training the weights ω_i on the new data subset ds_{i+1}. This method allows,

Table 2
Baseline results achieved by training from scratch the standard versions of YOLOv5, YOLOv8 and YOLOv11 with different densities. Results are reported in terms of mAP@0.5, $F1$ score, and inference speed.

Family	Size	mAP@0.5(%)	F1 Score (%)	Speed (ms, CPU)
v5	S	56.9 ± 0.2	60.2	299 ± 12.7
v5	M	55.8 ± 0.4	60.2	581.2 ± 23.8
v5	L	56.4 ± 0.4	60.8	1057.0 ± 55.0
v8	S	57.3 ± 0.4	61.3	**285.6 ± 20.6**
v8	M	57.3 ± 0.3	61.9	652.8 ± 36.9
v8	L	58.0 ± 0.1	61.9	1163.9 ± 2.3
v11	S	55.8 ± 0.2	60.2	291.3 ± 8.5
v11	M	**58.7 ± 1.0**	**62.6**	690.6 ± 45.5
v11	L	56.2 ± 0.6	60.9	875.6 ± 65.9

therefore, to improve the capabilities of the model to retain and adapt knowledge.

2.6. Experimental setup

As for the experimental setup, training and validation were performed using a DGX Station equipped with eight NVIDIA H100 GPUs, each with 96 GB of RAM. To evaluate the effectiveness of the approach on systems deployed directly in the field, possibly on self-moving robots, the inference speed was computed on an Intel Core i7 processor deployable on a small-sized NUC. All the source code was implemented using the Ultralytics library [9] with Python 3.12. Finally, it is important to underline how training and validation tests were performed on the *in-lab* dataset, while the *in-field* dataset was mainly used for the final domain adaptation via fine-tuning.

As for hyperparameter selection, the optimal setup was found for training and validation, optimising mAP@0.5, and exploiting the Ray-Tune library [11] for implementing the Hyperband optimisation algorithm [10].

3. Experimental results

In this section, the results achieved by the proposed approach are discussed.

3.1. Baseline

The first set of experiments was aimed at establishing a baseline of the results achieved using the base YOLOv5, YOLOv8, and YOLOv11 algorithms. To provide a fair comparison, all the proposed architectures were trained from scratch, not considering pretrained MS-COCO weights. For every model, all five densities, from *nano* to *extra-large*, were evaluated. Results for the three most well-performing densities are shown in Table 2; full results are reported in Appendix A. As expected, the best-performing model belongs to YOLOv11m, with an mAP@0.5 of 58.7 ± 1.0%, and an F1 score of 62.6%. Therefore, this model will be selected as the base for the ablation study detailed in the next Section.

3.2. Ablation study

To assess the impact of the attention mechanism within the YOLOv11m neck, an ablation study, considering the effects of SA and CBAM, along with the combined effect of both mechanisms, was proposed. To provide further context, the results reported in Table 3 also highlight the impact on YOLOv11s, whereas the full report is available in Appendix A.

The most interesting aspect highlighted by the ablation study is that the introduction of SA mechanisms improves the results when the small version of YOLO is considered, with a positive absolute difference of 1.6%, at the cost of a non-negligible impact in terms of inference speed

Table 3
Impact of attention mechanisms on the base YOLOv11 model. The changes in mAP@0.5 and inference speed are reported as an absolute percentage difference.

Model	mAP@0.5(%)	Diff (%)	Speed (ms, CPU)	Diff (%)
YOLOv11s	55.8 ± 0.4	-	291.3 ± 8.5	-
+ SA	57.4 ± 0.7	+ 1.6	424.0 ± 24.1	+ 45.6
+ CBAM	55.8 ± 0.4	0.0	527.6 ± 52.6	+ 81.1
+ CBAM + SA	57.4 ± 0.8	+ 1.6	478.2 ± 17.3	+ 67.3
YOLOv11m	58.7 ± 1.0	-	690.6 ± 45.5	-
+ SA	58.0 ± 0.6	- 0.7	1190.7 ± 54.0	+ 72.4
+ CBAM	58.2 ± 0.1	- 0.5	1450.6 ± 52.4	+ 110.1
+ CBAM + SA	57.9 ± 0.2	- 0.8	1270.1 ± 35.7	+ 83.9

Table 4
Results achieved by training the YOLOv11m network with an incremental learning approach.

Training method	mAP@0.5 (%)	F1 Score (%)	Speed (ms, CPU)
Standard	58.7 ± 1.0	62.6	690.6 ± 45.5
Incremental	60.06 ± 0.2	63.7	671.7 ± 16.4

of 45.6%. The CBAM mechanism does not yield relevant improvements, while still having an impact on inference speed, as further demonstrated by the combined action of both attention mechanisms.

As for the selected YOLOv11m baseline, the impacts of the addition of attention mechanisms resulted in a net degradation of performance both in terms of mAP and inference speed.

Therefore, for the task at hand, considering the specific in-lab dataset, it can be concluded that the insertion of attention mechanisms within YOLOv11m does not provide relevant improvements; therefore, the baseline YOLOv11m architecture can be used for the evaluation of the impacts of incremental learning in the next Section.

3.3. Impact of incremental learning

Table 4 reports the results achieved when the incremental learning was considered using YOLOv11m as the base architecture. The impact of using incremental learning is extremely interesting and can be quantified in three aspects. First, there is a noticeable improvement in terms of quantitative metrics, such as mAP@0.5 and F1 Score, of about 1.36 and 1.1%, respectively. Second, the inference speed is slightly improved when incremental learning is considered. Finally, results appear to be much more stable, given the significant reduction in variance shown.

3.4. Domain adaptation via fine-tuning

To demonstrate the capability of generalising between loosely coupled domains, the incremental model discussed in Section 3.3 was fine-tuned for domain adaptation and validated on the *in-field* dataset. In other words, the richer feature representation learned from the extensive in-lab data was used as the basis for fine-tuning for the specific in-field conditions.

After the fine-tuning process, the model was able to achieve 60.45% precision, 56.67% recall, and 55.95 mAP@0.5 on the problem posed by the in-field dataset, that is, the identification of ripe (red) and unripe (other) berries.

Some examples of the results are shown in Fig. 8. From the confusion matrix, shown in Fig. 11, it is clear that the model achieves 85.56% score of red berry identification with respect to the labelled ground truth, even if there is a relatively high number of background false positives.

4. Discussion

The experimental results described in Section 3 highlighted the positive outcome of the proposed Incremental Learning over the standard

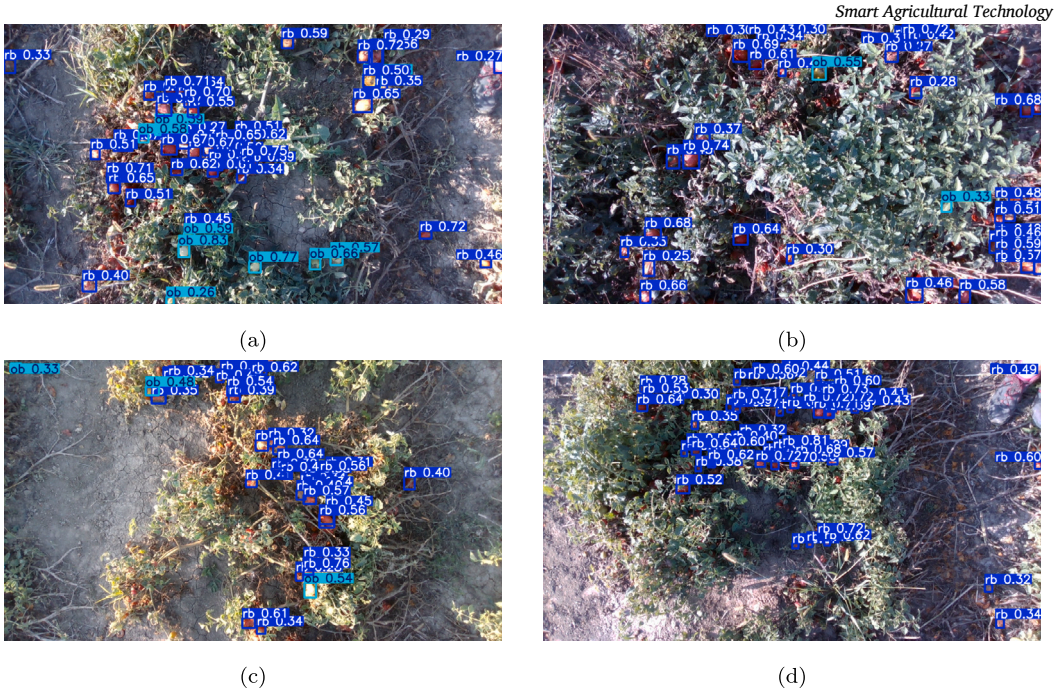

Fig. 8. Results achieved by using fine tuning for domain adaptation on the in-field dataset with the best-performing incremental learning model.

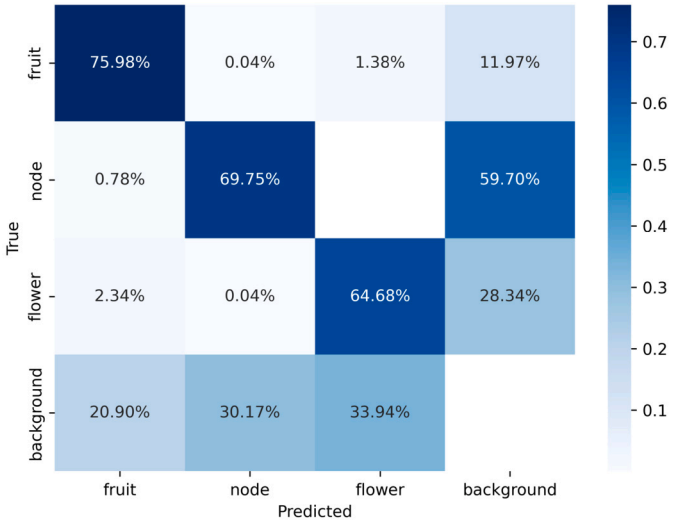

Fig. 9. Confusion matrix for the baseline experiment using YOLOv11x with high resolution.

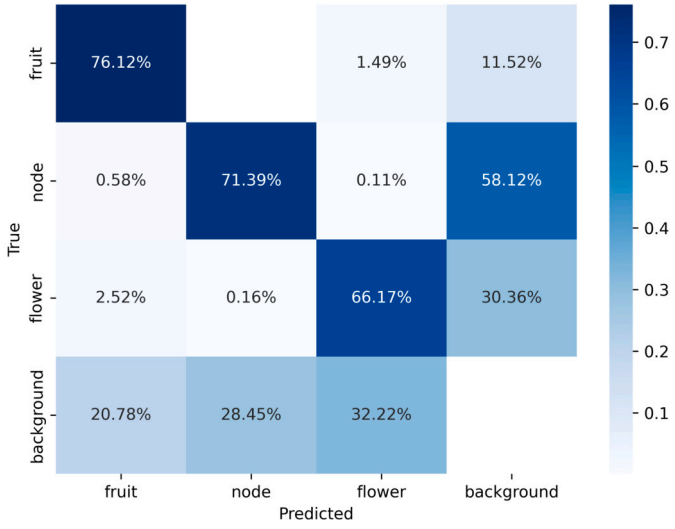

Fig. 10. Confusion matrix for Incremental learning using YOLOv11m.

training procedure for YOLOv11m. Still, it is worth discussing some interesting aspects which were revealed by the investigation.

4.1. Effectiveness of incremental learning

Let us start with the impact of Incremental Learning on the baseline model. It is important to underline that the baseline model is exposed, during training, to the whole training set. This implies that the model will be more likely to learn features which minimise loss on this set and, therefore, are able to capture generic relationships and characteristics within the data. While desirable, this effect may impact the decision capabilities of the model in certain specific situations, therefore, undermining its global performance. This is also demonstrated, on the case at hand, by the high variability ($\pm 1.0\%$) demonstrated by the baseline model. The Incremental Learning process mitigates this effect, exposing the model only to a smaller, and possibly more cohesive, data sample at each incremental step. By using this solution, the model is able to provide more effective and stable predictions, resulting in an overall increase of the provided metrics in terms of value and stability. In other words, the Incremental Learning steps can be seen as successive fine-tunings of the model, allowing it to retain generalisation capabilities, but continuously improving its performance on smaller and less present details in some subsets of the dataset.

4.2. Applicability of attention mechanisms

Another counterintuitive yet critical finding is that the effectiveness of attention mechanics is strictly related to the capacity of the baseline model. In other words, the performance of the model with attention mechanisms is strictly related to the intrinsic capacities of its corresponding baseline. Specifically, for the smaller-density smaller YOLOv11s model, the insertion of attention mechanisms provides the most noticeable effects if compared to the larger YOLOv11m, following

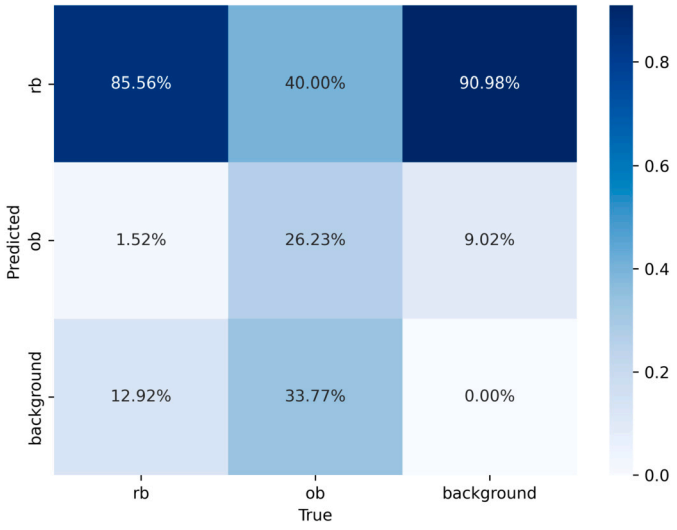

Fig. 11. Confusion matrix for the in-field dataset using fine-tuning for domain adaptation.

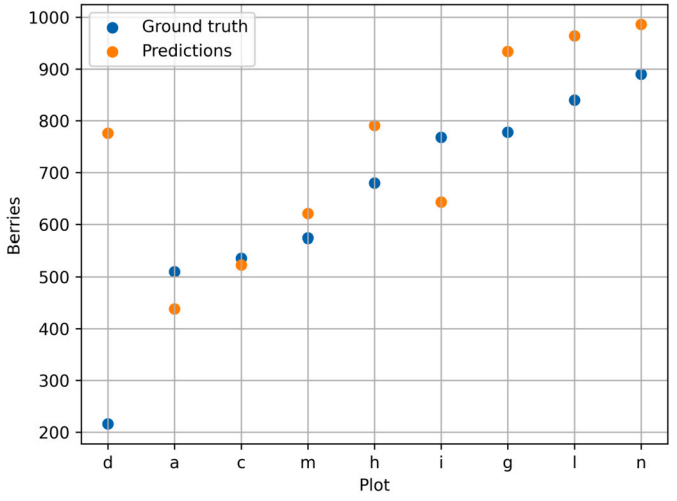

Fig. 12. Comparison of automated berry counting with ground-truth counting provided by farmers in different tomato plots, with plots *d*, *m*, *h* referring to San Marzano tomato variety, *a*, *c*, *l* to UC 82 and *i*, *g*, *n* to Mariner F1.

the established trend that attention provides more relevant improvements on smaller architectures [19,7]. This is also related to the higher number of parameters resulting from the insertion of attention layers, which would require a significantly higher number of data samples for effective training. However, the improvements provided by attention mechanisms to YOLOv11s are still not enough to outmatch the performance of unmodified YOLOv11m. Furthermore, attention mechanisms also cause severe penalties to inference speeds, undermining the applicability of the models in real-time settings. These aspects suggest that YOLOv11m represents, in its core version, a "sweet spot" for the task at hand, with an optimal model capacity for the specific experiment.

4.3. Domain adaptation and practical implications

Another relevant finding was the effectiveness of domain adaptation, which involved fine-tuning the model resulting from incremental learning on the *in-lab* dataset to the *in-field* context. The effectiveness of the adaptation, demonstrated by the mAP@0.5 score of 55.95% achieved despite the extremely limited size of the in-field dataset, suggests the viability of a two-stage adaptation pipeline, where models trained using Incremental Learning on a controlled context can adapt their weights via fine-tuning to loosely coupled settings, retaining robustness and avoiding large data gathering and labelling procedures to be taken directly on the field. The effectiveness is also demonstrated by Fig. 12, where the predictions made by the network after fine-tuning are compared against the ground truth manually counted by the experts. Interestingly, it can be seen that the network provides consistent results with manual counting, showing significant differences only in one case (plot *d*), implying some misbehaviour in the data collection or preprocessing procedure.

4.4. Limitations and drawbacks

Still, the experiments highlighted several shortcomings and challenges which remain to be addressed. First, let us start by analysing the confusion matrix for the YOLOv11m baseline model, shown in Fig. 9, and compare it with the one for Incremental Learning, shown in Fig. 10. Let us recall that the matrix considered for object detection reports also background false positives, that is, detections which were not explicitly labelled by domain experts, as well as background false negatives, that is, labelled objects which were not identified by the network during the evaluation. When Incremental Learning is applied, correct predictions noticeably improve by 0.14% for fruit, 1.64% for flowers, and by 1.49% for nodes. At the same time, the impact of background false negatives is reduced by 0.12% for fruit, 1.72% for flowers, and also by 1.72% for nodes. Still, when background false positives are considered, a noticeable increment is in terms of flowers (2.02%), with a reduction of fruit (0.45%) and nodes (1.58%). While encouraging, the large number of background false positives is directly related to the aspects already described in Cardellicchio et al. [3], with the models yielding several background false positives for nodes, and significantly lower amounts for fruit and flowers. To discuss such effects, let us start with the following considerations for the labelling process.

- For fruit, most of the plants shown in the in-lab dataset were represented during the growth stage; therefore, unripe berries were also visible, but not labelled by domain experts, as not deemed relevant in the phenotypical analysis.
- For nodes, most of the labels provided by domain experts were on the main stem, at this was the most important aspect identified by domain experts.
- For flowers, most of the items were extremely small and easily missed during labelling.

Given these starting points, it was clear how a large part of the background positives could be qualitatively explained as follows.

- As for fruit, the model identified many unripe berries.
- As for nodes, the model identified many of them on secondary stems.
- As for flowers, the model was able to identify small flowers which domain experts missed during labelling.

These three effects are mainly related to the fact that all these models use only visual information to provide localisation and classification. That is, there is no topological or contextual information which allows the model to distinguish between nodes on the primary or on the secondary stems, for example. Of these three effects, only the third is desirable from the phenotypical perspective, as the model is overcoming the limitations provided by the eye of the human expert. As for the other two, they add noise to the achieved results; therefore, future iterations of the model should deal with this issue, for example, by inserting topological information on the structure of the plants, thus allowing a proper processing pipeline to select only nodes on the main stem.

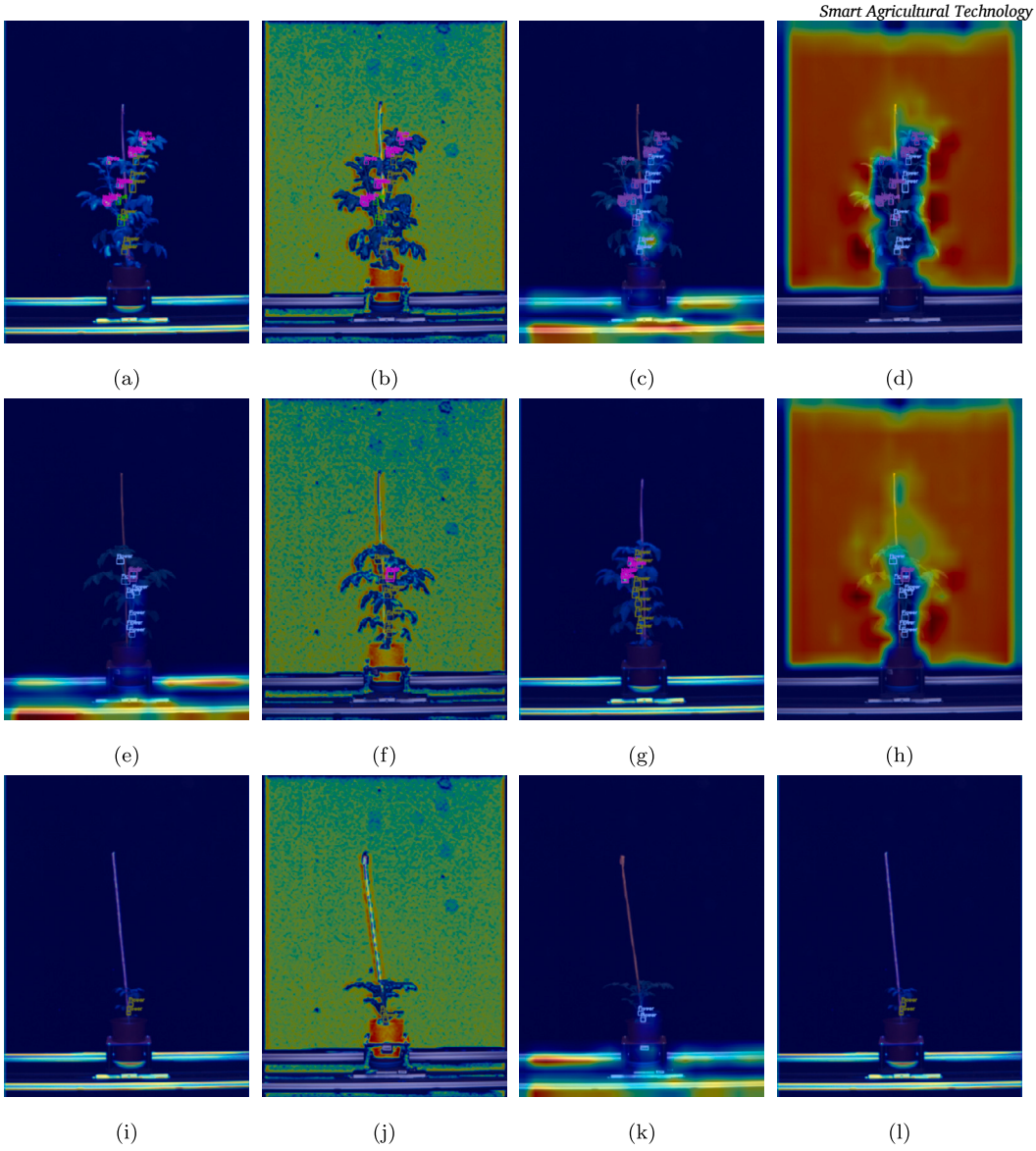

Fig. 13. Class Activation Maps of the backbone computed using EigenCAM. Each row represents a specific image, while each column represents the CAMs extracted from a certain layer. Specifically, the first column represents the CAMs extracted from the second convolution layer, the second column the CAMs from the first C3K2 layer, while the third and the fourth columns represent the CAMs extracted from the latest convolution and C3K2 layers, respectively. These CAMs were extracted for the baseline version of YOLOv11x.

4.5. Results interpretability

To properly highlight the decisions provided by the model, the class activation maps achieved via EigenCAM were used to visualise parts of the image considered by the model in performing its decisions. Specifically, the activation maps were extracted from the two initial C3K2 layers of the backbone to assess the evolution of the gradients during the first stages of feature detection and the latest C3K2 layers of the neck, after feature fusion, and immediately before feeding the detection head. Three specific samples were selected according to the density and the growth stage of the plant, specifically, an adult plant with high density of phenotypical traits (Figs. 13-15, first row), a medium density (Figs. 13-15, second row), and a low density (Figs. 13-15, third row). The results were evaluated for the baseline model of YOLOv11m (Fig. 13), the Incremental Learning mode for YOLOv11m at its first iteration (Fig. 14), and the Incremental Learning mode at the end of the final increment (Fig. 15). Let us start by evaluating the baseline model, as in Fig. 13. In this case, the network starts without any relevant information for object detection, and tends to focus its attention on the table (first column) and, immediately after, on the most relevant edges. Then, after feature fusion, the network shows varying behaviours: it focuses on the background in the case of medium and high phenotypical traits, and on the support in the case of low phenotypical traits. This suggests less reliability of the network, which may often focus on non-relevant patches of the image. When incremental learning is involved, the network starts immediately by focusing on edges, probably due to the lack of information concerning the support in the background. Then, after feature fusion, the network shifts its attention to the relevant part of the image, regardless of the density, as shown in Fig. 14. Finally, when the overall Incremental Learning procedure is complete, the network shows the immediate capability of shifting its focus from the background, looking for the overall context of the image, to the relevant object, already in the first backbone layers. Then, as in the initial step, it shifts its attention to relevant phenotypical traits, including also, in this case, a stronger focus on the supporting table, which is, however, not considered during detection. This suggests that the Incremental Learning procedure forces

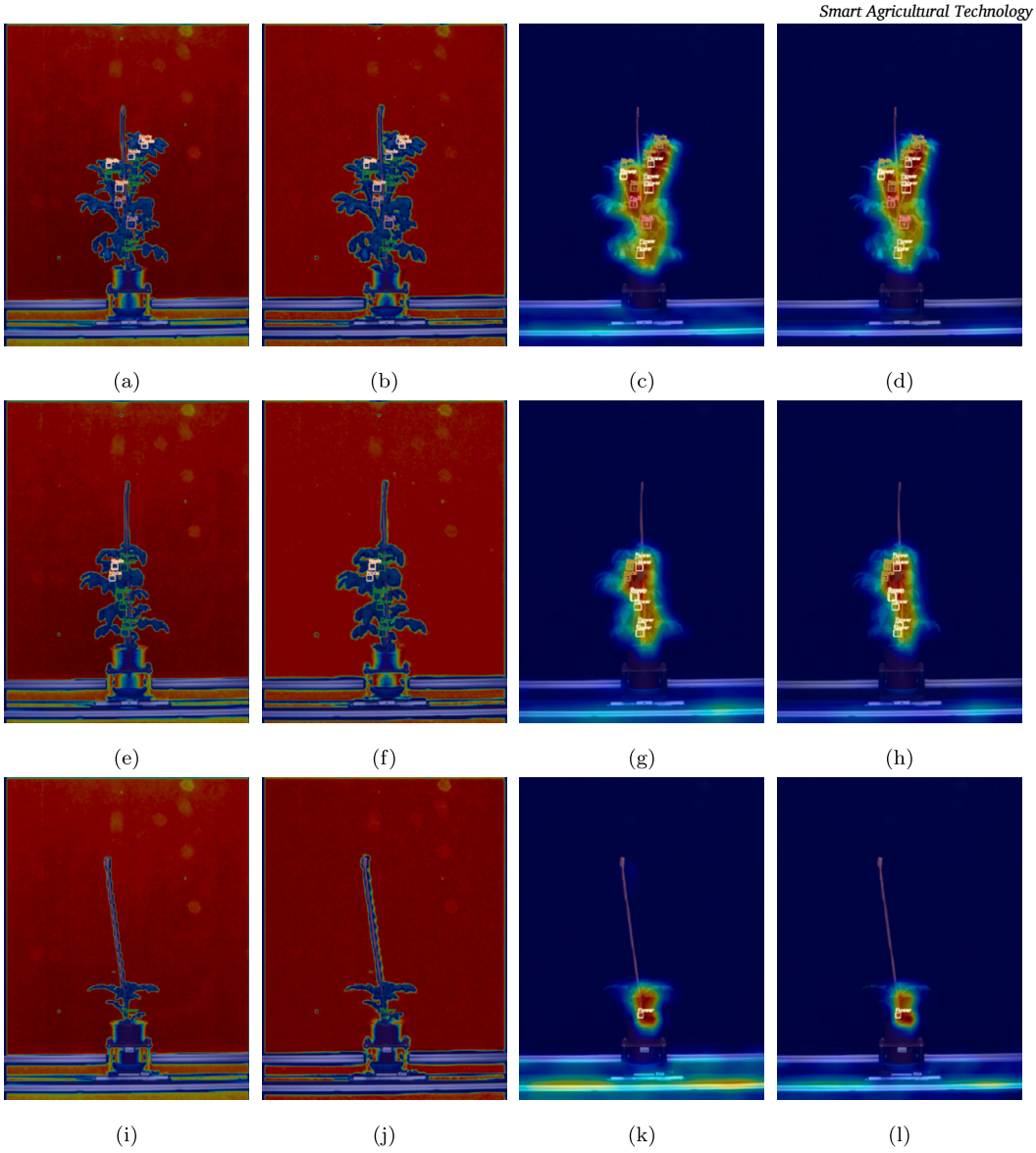

Fig. 14. Class Activation Maps of the backbone computed using EigenCAM. Each row represents a specific image, while each column represents the CAMs extracted from a certain layer. Specifically, the first column represents the CAMs extracted from the second convolution layer, the second column the CAMs from the first C3K2 layer, while the third and the fourth columns represent the CAMs extracted from the latest convolution and C3K2 layers, respectively. These CAMs were extracted for the initial step of the federated version of YOLOv11m.

the network to focus on relevant details even in early iterations, and forces it to provide a more effective representation of the sample under investigation.

5. Conclusions and future works

This work proposed a refined Incremental Learning strategy for plant phenotyping in tomato crops. Specifically, the work highlighted how the YOLOv11m single-stage object detector outperforms the other members of its family, thus providing a sweet spot for the task at hand. Furthermore, it was demonstrated how the insertion of attention mechanisms may not always provide improved results, as these are highly dependent on the density of the original model and the overall extension of the dataset. To provide more robust performance, the Incremental Learning strategy was applied to the baseline YOLOv11m model, achieving relevant improvements in both quantitative metrics, with an increase of 1.36% in terms of mAP@0.5, and 1.2% in terms of F1 score, and computing efficiency, with a reduction of 19 ms in inference speed. Domain adaptation was also demonstrated on small datasets by using fine-tuning.

Several limitations still remain, mainly due to the dataset itself. First, current labelling shows flaws, specifically in the *in-field* dataset, which produces a high number of background false positives. This should be properly addressed in future versions. Furthermore, it is important to underline how the limited cardinality of even the *in-lab* dataset undermines the applicability of attention mechanisms. Finally, to provide a proper integration of attention mechanism in an effective fashion, further and more extensive modifications to the base networks should be proposed, optimising the computational load and the number of parameters to allow the network to improve achieved results while still retaining competitive quantitative performance.

Overall, this work provides a robust proposal based on Incremental Learning to improve the effectiveness of baseline models, which could be potentially applied even to other unrelated fields, and yield practical advancements in the application of deep learning object detectors in plant phenotyping.

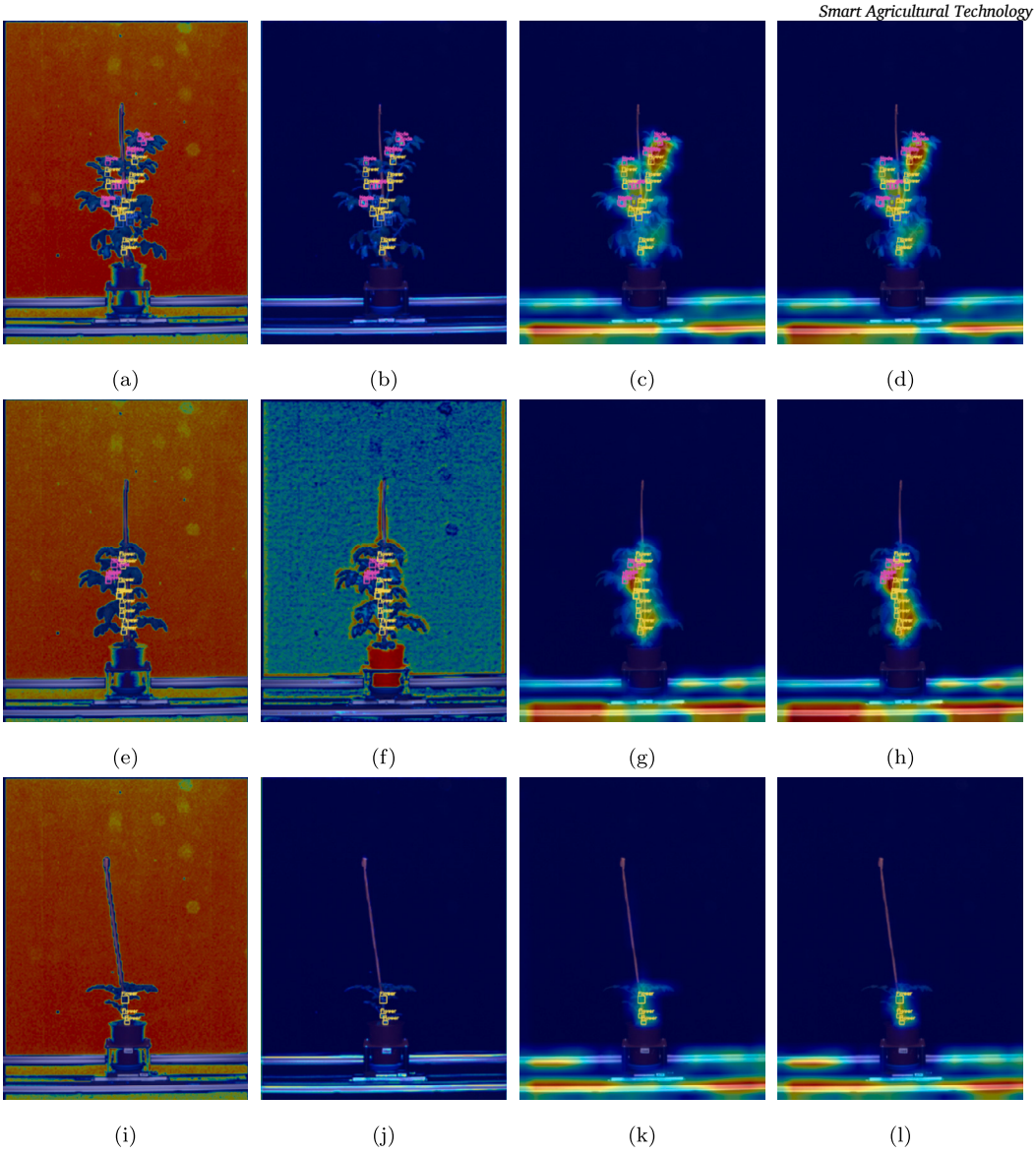

Fig. 15. Class Activation Maps of the backbone computed using EigenCAM. Each row represents a specific image, while each column represents the CAMs extracted from a certain layer. Specifically, the first column represents the CAMs extracted from the second convolution layer, the second column the CAMs from the first C3K2 layer, while the third and the fourth columns represent the CAMs extracted from the latest convolution and C3K2 layers, respectively. These CAMs were extracted for the last fine-tuned step of the Incremental Learning version of YOLOv11m.

CRediT authorship contribution statement

Angelo Cardellicchio: Writing – original draft, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Vito Renò:** Writing – original draft, Validation, Supervision, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Francesco Cellini:** Visualization, Resources, Data curation. **Stephan Summerer:** Visualization, Resources, Investigation, Data curation. **Angelo Petrozza:** Visualization, Resources, Investigation, Data curation. **Annalisa Milella:** Writing – review & editing, Supervision, Methodology, Formal analysis, Conceptualization.

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.

Acknowledgements

This work was partially funded by the projects: AgRibot-Harnessing Robotics, XR/AR, and 5G for a New Era of Safe, Sustainable, and Smart Agriculture, European Union's Horizon Europe research and innovation programme under grant agreement (No. 101183158); E-crops - Technologies for Digital and Sustainable Agriculture, Italian Ministry of University and Research (MUR) under the PON Agrifood Program (No. ARS01_01136); CNR DIITET project DIT.AD022.207, STRIVE-le Scienze per le TRansizioni Industriale, Verde ed Energetica (FOE 2022), sub-task activity Agro-Sensing2. The authors are grateful to all project partners contributing to the experimental setup and design and to ground-truth data gathering.

Appendix A. Experimental report

In this Appendix, a full report of the performed experiments is provided. Each result, apart from F1 scores, is reported as the arithmetic

Table A.5
Full baseline results for detection accuracy (mAP scores) and inference speed.

Model	mAP@0.5 (%)	mAP@.5-.95 (%)	Speed (ms)
v5n	52.10 ± 0.21	17.93 ± 0.06	166.0 ± 15.3
v5s	56.85 ± 0.17	20.50 ± 0.07	299.9 ± 12.7
v5m	55.77 ± 0.45	20.15 ± 0.30	581.2 ± 23.8
v5l	56.44 ± 0.42	20.42 ± 0.34	1057.0 ± 55.0
v5x	54.98 ± 0.46	19.73 ± 0.18	1718.1 ± 34.8
v8n	53.27 ± 0.17	18.74 ± 0.10	165.8 ± 23.9
v8s	57.25 ± 0.42	20.22 ± 0.10	285.6 ± 20.6
v8m	57.28 ± 0.29	20.83 ± 0.20	652.8 ± 36.9
v8l	58.04 ± 0.13	21.09 ± 0.19	1163.9 ± 2.3
v8x	56.91 ± 0.34	20.06 ± 0.28	1823.4 ± 103.3
v11n	52.11 ± 0.30	18.10 ± 0.19	168.8 ± 32.4
v11s	55.82 ± 0.36	19.54 ± 0.15	291.3 ± 8.5
v11m	58.70 ± 0.96	21.51 ± 0.42	690.6 ± 45.5
v11l	56.20 ± 0.64	20.07 ± 0.23	875.6 ± 65.9
v11x	57.30 ± 0.60	20.71 ± 0.28	1564.1 ± 21.8

Table A.6
Full baseline results for classification metrics (Precision, Recall, F1 Score).

Model	Precision (%)	Recall (%)	F1 Score (%)
v5n	57.51 ± 0.25	56.62 ± 0.21	57.05
v5s	60.15 ± 0.54	60.23 ± 0.45	60.19
v5m	60.40 ± 0.51	60.09 ± 0.55	60.22
v5l	60.90 ± 1.09	60.72 ± 0.53	60.81
v5x	60.31 ± 0.28	60.57 ± 0.64	60.44
v8n	58.58 ± 0.56	57.22 ± 0.18	57.89
v8s	63.83 ± 0.86	58.94 ± 0.28	61.26
v8m	62.14 ± 0.49	61.63 ± 0.38	61.89
v8l	62.40 ± 0.29	61.40 ± 0.09	61.90
v8x	61.93 ± 0.73	62.52 ± 0.55	62.22
v11n	56.45 ± 0.35	57.69 ± 0.41	57.06
v11s	59.45 ± 0.50	60.99 ± 0.69	60.21
v11m	62.31 ± 1.24	63.00 ± 0.63	62.64
v11l	61.46 ± 0.68	60.42 ± 0.81	60.90
v11x	61.04 ± 1.07	61.45 ± 0.38	61.24

Table A.7
Full results for models with CBAM attention, showing detection accuracy (mAP scores) and inference speed.

Model	mAP@0.5 (%)	mAP@0.5-0.95 (%)	Speed (ms, CPU)
v5n+CBAM	53.59 ± 0.13	18.58 ± 0.08	223.1 ± 3.9
v5s+CBAM	57.39 ± 0.34	20.74 ± 0.16	529.1 ± 51.9
v5m+CBAM	56.88 ± 0.09	20.70 ± 0.06	1197.1 ± 68.9
v5l+CBAM	56.86 ± 0.17	21.05 ± 0.08	1823.1 ± 54.3
v5x+CBAM	54.45 ± 0.66	19.16 ± 0.37	3232.2 ± 202.0
v8n+CBAM	55.31 ± 0.13	19.80 ± 0.03	308.1 ± 46.8
v8s+CBAM	58.05 ± 0.16	21.48 ± 0.12	738.2 ± 60.3
v8m+CBAM	58.11 ± 0.44	21.25 ± 0.08	1529.5 ± 75.9
v8l+CBAM	56.14 ± 0.42	20.14 ± 0.19	2841.4 ± 178.4
v8x+CBAM	54.89 ± 0.30	19.34 ± 0.16	4144.4 ± 66.8
v11n+CBAM	54.35 ± 0.89	19.18 ± 0.23	213.7 ± 47.7
v11s+CBAM	55.84 ± 0.41	20.12 ± 0.22	527.6 ± 52.6
v11m+CBAM	58.20 ± 0.14	21.49 ± 0.24	1450.6 ± 52.4
v11l+CBAM	56.69 ± 0.34	20.73 ± 0.25	1722.0 ± 57.9
v11x+CBAM	56.54 ± 0.59	20.76 ± 0.22	3190.1 ± 64.4

mean ± standard deviation from 3 independent training runs. (See Tables A.5–A.12.)

Table A.8
Full results for models with CBAM attention, showing classification metrics (Precision, Recall, F1 Score).

Model	Precision (%)	Recall (%)	F1 Score (%)
v5n+CBAM	58.72 ± 0.39	57.11 ± 0.47	57.89
v5s+CBAM	61.72 ± 0.68	59.81 ± 0.29	60.74
v5m+CBAM	61.37 ± 0.52	60.03 ± 0.26	60.68
v5l+CBAM	61.30 ± 0.50	61.30 ± 0.12	61.30
v5x+CBAM	60.12 ± 0.30	60.89 ± 0.67	60.50
v8n+CBAM	60.52 ± 0.50	58.15 ± 0.25	59.30
v8s+CBAM	61.75 ± 0.69	60.84 ± 0.11	61.29
v8m+CBAM	64.08 ± 0.94	61.54 ± 0.43	62.77
v8l+CBAM	60.78 ± 0.85	60.72 ± 1.06	60.74
v8x+CBAM	61.24 ± 0.95	59.97 ± 0.45	60.59
v11n+CBAM	60.20 ± 1.17	57.81 ± 0.34	58.96
v11s+CBAM	59.89 ± 0.55	60.53 ± 0.36	60.20
v11m+CBAM	61.30 ± 0.72	63.32 ± 0.81	62.24
v11l+CBAM	60.12 ± 0.55	61.29 ± 0.77	60.70
v11x+CBAM	60.18 ± 0.30	62.30 ± 0.18	61.21

Table A.9
Full results for models with SA attention, showing detection accuracy (mAP scores) and inference speed.

Model	mAP@0.5 (%)	mAP@0.5-0.95 (%)	Speed (ms, CPU)
v5n+SA	50.19 ± 0.43	17.28 ± 0.17	169.4 ± 18.8
v5s+SA	55.43 ± 0.34	19.82 ± 0.14	336.9 ± 24.2
v5m+SA	57.24 ± 0.35	20.29 ± 0.26	738.0 ± 24.6
v5l+SA	57.31 ± 0.32	20.46 ± 0.16	1381.0 ± 2.4
v5x+SA	55.78 ± 0.23	19.90 ± 0.13	2274.3 ± 149.4
v8n+SA	55.59 ± 0.21	19.73 ± 0.06	240.0 ± 34.8
v8s+SA	57.11 ± 0.11	20.57 ± 0.11	633.6 ± 33.8
v8m+SA	56.46 ± 0.30	20.52 ± 0.24	1365.3 ± 16.9
v8l+SA	57.29 ± 0.65	20.73 ± 0.32	2230.9 ± 208.7
v8x+SA	56.66 ± 0.11	19.62 ± 0.07	3620.6 ± 53.1
v11n+SA	53.65 ± 0.49	18.63 ± 0.13	177.7 ± 7.7
v11s+SA	57.42 ± 0.75	21.12 ± 0.46	424.0 ± 24.1
v11m+SA	58.00 ± 0.58	21.27 ± 0.21	1190.7 ± 54.0
v11l+SA	56.76 ± 0.22	20.91 ± 0.15	1484.4 ± 155.1
v11x+SA	56.26 ± 0.19	20.57 ± 0.21	2520.0 ± 42.9

Table A.10
Full results for models with SA attention, showing classification metrics (Precision, Recall, F1 Score).

Model	Precision (%)	Recall (%)	F1 Score (%)
v5n+SA	55.35 ± 0.14	55.57 ± 0.61	55.46
v5s+SA	60.56 ± 0.23	58.91 ± 0.51	59.72
v5m+SA	60.55 ± 0.34	60.29 ± 0.60	60.42
v5l+SA	62.00 ± 0.39	61.75 ± 0.42	61.87
v5x+SA	60.91 ± 0.14	61.34 ± 0.52	61.12
v8n+SA	60.72 ± 0.58	58.64 ± 0.21	59.65
v8s+SA	61.01 ± 0.44	61.33 ± 0.24	61.17
v8m+SA	59.61 ± 0.49	61.82 ± 0.21	60.68
v8l+SA	61.56 ± 0.75	63.19 ± 0.50	62.36
v8x+SA	62.13 ± 0.19	63.09 ± 0.23	62.60
v11n+SA	58.06 ± 0.44	57.32 ± 0.55	57.72
v11s+SA	61.71 ± 1.23	60.80 ± 0.32	61.26
v11m+SA	61.61 ± 0.81	62.62 ± 0.42	62.11
v11l+SA	60.73 ± 0.70	61.35 ± 0.70	61.06
v11x+SA	61.19 ± 0.60	61.72 ± 0.61	61.42

Table A.11
Full results for models with combined CBAM & SA attention, showing detection accuracy (mAP scores) and inference speed.

Model	mAP@0.5 (%)	mAP@0.5-0.95 (%)	Speed (ms, CPU)
v5n+CBAM&SA	52.46 ± 0.34	18.29 ± 0.13	198.3 ± 13.9
v5s+CBAM&SA	57.40 ± 0.44	20.67 ± 0.17	487.2 ± 17.3
v5m+CBAM&SA	56.76 ± 0.45	20.37 ± 0.17	1270.1 ± 35.7
v5l+CBAM&SA	57.19 ± 0.26	20.61 ± 0.14	1392.2 ± 38.6
v5x+CBAM&SA	57.06 ± 0.22	20.57 ± 0.22	2854.1 ± 173.2
v8n+CBAM&SA	52.46 ± 0.34	18.29 ± 0.13	198.3 ± 13.9
v8s+CBAM&SA	57.40 ± 0.44	20.67 ± 0.17	487.2 ± 17.3
v8m+CBAM&SA	56.76 ± 0.45	20.37 ± 0.17	1270.1 ± 35.7
v8l+CBAM&SA	57.19 ± 0.26	20.61 ± 0.14	1392.2 ± 38.6
v8x+CBAM&SA	57.06 ± 0.22	20.57 ± 0.22	2854.1 ± 173.2
v11n+CBAM&SA	53.67 ± 0.63	18.80 ± 0.33	198.3 ± 13.9
v11s+CBAM&SA	57.35 ± 0.79	20.58 ± 0.37	487.2 ± 17.3
v11m+CBAM&SA	57.95 ± 0.20	21.20 ± 0.10	1270.1 ± 35.7
v11l+CBAM&SA	56.96 ± 0.66	20.75 ± 0.27	1392.2 ± 38.6
v11x+CBAM&SA	55.47 ± 0.64	20.40 ± 0.29	2854.1 ± 173.2

Table A.12
Full results for models with combined CBAM & SA attention, showing classification metrics (Precision, Recall, F1 Score).

Model	Precision (%)	Recall (%)	F1 Score (%)
v5n+CBAM&SA	56.41 ± 0.50	57.28 ± 0.28	56.84
v5s+CBAM&SA	60.51 ± 0.41	61.32 ± 0.47	60.91
v5m+CBAM&SA	59.90 ± 0.50	61.30 ± 0.36	60.59
v5l+CBAM&SA	60.91 ± 0.48	61.52 ± 0.51	61.21
v5x+CBAM&SA	60.65 ± 0.35	61.78 ± 0.29	61.21
v8n+CBAM&SA	56.41 ± 0.50	57.28 ± 0.28	56.84
v8s+CBAM&SA	60.51 ± 0.41	61.32 ± 0.47	60.91
v8m+CBAM&SA	59.90 ± 0.50	61.30 ± 0.36	60.59
v8l+CBAM&SA	60.91 ± 0.48	61.52 ± 0.51	61.21
v8x+CBAM&SA	60.65 ± 0.35	61.78 ± 0.29	61.21
v11n+CBAM&SA	57.75 ± 1.37	57.79 ± 0.27	57.77
v11s+CBAM&SA	60.45 ± 0.96	60.72 ± 0.26	60.58
v11m+CBAM&SA	61.58 ± 0.37	61.84 ± 0.40	61.71
v11l+CBAM&SA	61.96 ± 0.74	61.35 ± 0.23	61.65
v11x+CBAM&SA	59.55 ± 0.23	61.47 ± 0.71	60.48

Data availability

It is important to underline that one of the most relevant limitations of the current works in the research community is the possibility of performing a thorough comparison between a proposal and other state-of-the-art models. This is mainly related to an extremely important aspect, that is, the need to provide access to the dataset used for the evaluation, along with the need to provide access to the models proposed for the advancement, which some authors cannot provide for copyright issues [4]. For this reason, several works propose ablation tests and comparisons with baseline models [22], highlighting the improvements in those terms. In this work, the authors decided to provide a similar perspective while discussing the achieved results; however, to allow other researchers to reproduce their results and provide transparency, they will make their dataset and trained models available upon request by the scientific community.

References

[1] M.L. Ali, Z. Zhang, The yolo framework: a comprehensive review of evolution, applications, and benchmarks in object detection, Computers 13 (2024) 336.
[2] D. Cammarano, S. Jamshidi, G. Hoogenboom, A.C. Ruane, D. Niyogi, D. Ronga, Processing tomato production is expected to decrease by 2050 due to the projected increase in temperature, Nat. Food 3 (2022) 437–444, https://doi.org/10.1038/s43016-022-00521-y.
[3] A. Cardellicchio, F. Solimani, G. Dimauro, A. Petrozza, S. Summerer, F. Cellini, V. Renò, Detection of tomato plant phenotyping traits using YOLOv5-based single stage detectors, Comput. Electron. Agric. 207 (2023) 107757, https://doi.org/10.1016/j.compag.2023.107757, https://www.sciencedirect.com/science/article/pii/S016816992300145X.
[4] W. Chen, M. Liu, C. Zhao, X. Li, Y. Wang, Mtd-yolo: multi-task deep convolutional neural network for cherry tomato fruit bunch maturity detection, Comput. Electron. Agric. 216 (2024) 108533, https://doi.org/10.1016/j.compag.2023.108533, https://www.sciencedirect.com/science/article/pii/S0168169923009213.
[5] C.T. Chien, R.Y. Ju, K.Y. Chou, E. Xieerke, J.S. Chiang, Yolov8-am: Yolov8 based on effective attention mechanisms for pediatric wrist fracture detection, IEEE Access (2025).
[6] G. Gao, C. Shuai, S. Wang, T. Ding, Using improved yolo v5s to recognize tomatoes in a continuous working environment, Signal Image Video Process. 18 (2024) 4019–4028.
[7] X. Guo, J. Wang, G. Gao, Z. Cheng, Z. Qiao, R. Zhang, Z. Ma, X. Wang, Lwheatnet: a lightweight convolutional neural network with mixed attention mechanism for wheat seed classification, Front. Plant Sci. 15 (2025) 1509656.
[8] G. Jocher, A. Haurasia, J. Qiu, Yolov8 by ultralytics, https://github.com/ultralytics/ultralytics, 2023.
[9] G. Jocher, J. Qiu, Ultralytics yolo11, https://github.com/ultralytics/ultralytics, 2024.
[10] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, A. Talwalkar, Hyperband: a novel bandit-based approach to hyperparameter optimization, J. Mach. Learn. Res. 18 (2018) 1–52.
[11] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J.E. Gonzalez, I. Stoica, Tune: a research platform for distributed model selection and training, arXiv preprint, arXiv:1807.05118, 2018.
[12] M. Ma, P.W.J. Taylor, D. Chen, N. Vaghefi, J.Z. He, Major soilborne pathogens of field processing tomatoes and management strategies, Microorganisms 11 (2023), https://doi.org/10.3390/microorganisms11020263, https://www.mdpi.com/2076-2607/11/2/263.
[13] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: unified, real-time object detection, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788.
[14] F. Solimani, A. Cardellicchio, G. Dimauro, A. Petrozza, S. Summerer, F. Cellini, V. Renò, Optimizing tomato plant phenotyping detection: boosting yolov8 architecture to tackle data complexity, Comput. Electron. Agric. 218 (2024) 108728, https://doi.org/10.1016/j.compag.2024.108728, https://www.sciencedirect.com/science/article/pii/S0168169924001194.
[15] F. Solimani, A. Cardellicchio, M. Nitti, A. Lako, G. Dimauro, V. Renò, A systematic review of effective hardware and software factors affecting high-throughput plant phenotyping, Information 14 (2023), https://doi.org/10.3390/info14040214, https://www.mdpi.com/2078-2489/14/4/214.
[16] X. Sun, Enhanced tomato detection in greenhouse environments: a lightweight model based on s-yolo with high accuracy, Front. Plant Sci. 15 (2024) 1451018.
[17] C.Y. Wang, H.Y.M. Liao, Y.H. Wu, P.Y. Chen, J.W. Hsieh, I.H. Yeh, Cspnet: a new backbone that can enhance learning capability of cnn, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 390–391.
[18] S. Wang, H. Jiang, J. Yang, X. Ma, J. Chen, Z. Li, X. Tang, Lightweight tomato ripeness detection algorithm based on the improved rt-detr, Front. Plant Sci. 15 (2024) 1415297.
[19] Y. Wang, W. Wang, Y. Li, Y. Jia, Y. Xu, Y. Ling, J. Ma, An attention mechanism module with spatial perception and channel information interaction, Complex Intell. Syst. 10 (2024) 5427–5444.
[20] S. Woo, J. Park, J.Y. Lee, I.S. Kweon, Cbam: convolutional block attention module, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 3–19.
[21] M. Wu, H. Lin, X. Shi, S. Zhu, B. Zheng, Mts-yolo: a multi-task lightweight and efficient model for tomato fruit bunch maturity and stem detection, Horticulturae 10 (2024) 1006.
[22] X. Yue, K. Qi, F. Yang, X. Na, Y. Liu, C. Liu, Rsr-yolo: a real-time method for small target tomato detection based on improved yolov8 network, Discov. Appl. Sci. 6 (2024) 268, https://doi.org/10.1007/s42452-024-05914-1.
[23] T. Zeng, S. Li, Q. Song, F. Zhong, X. Wei, Lightweight tomato real-time detection method based on improved yolo and mobile deployment, Comput. Electron. Agric. 205 (2023) 107625, https://doi.org/10.1016/j.compag.2023.107625, https://www.sciencedirect.com/science/article/pii/S0168169923000133.
[24] Q.L. Zhang, Y.B. Yang, Sa-net: shuffle attention for deep convolutional neural networks, in: ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2021, pp. 2235–2239.
[25] T. Zhang, L. Zhong, S. Wu, K. Han, B. Peng, Bcp-yolo: a lightweight industrial tomato detection method for uav inspection, in: Proceedings of the 2024 7th International Conference on Computer Information Science and Artificial Intelligence, 2024, pp. 474–480.
[26] J. Zhao, C. Du, Y. Li, M. Mudhsh, D. Guo, Y. Fan, X. Wu, X. Wang, R. Almodfer, Yolo-Granada: a lightweight attentioned yolo for pomegranates fruit detection, Sci. Rep. 14 (2024) 16848.