cocoaDeep: a preliminary study of the performance sensitivity to datasets of Faster RCNN, YOLO and transformer networks for cocoa pod detection

Philippe Borianne, Frédéric Théveny, Llorenç Cabrera-Bosquet, Sabine-Karen Lammoglia

cocoaDeep: a preliminary study of the performance sensitivity to datasets of Faster RCNN, YOLO and transformer networks for cocoa pod detection.

Philippe Borianne[1,2], Frédéric Théveny[1,2], Llorenç Cabrera-Bosquet[3], Sabine-Karen Lammoglia[4,5].

[1]CIRAD, UMR AMAP, F-34398 Montpellier, France; [2]AMAP, Univ Montpellier, CIRAD, CNRS, INRAE, IRD, Montpellier, France; [3]LEPSE, Univ Montpellier, INRAE, Montpellier Institut Agro, Montpellier, France; [4]UFR Biosciences, Univ Félix Houphouët-Boigny, Abidjan, Côte d'Ivoire; [5]CIRAD, UMR ABSYS, Montpellier, France.

Keywords

cocoa pod detection, low-resolution images, neural networks, YOLO, Transformers, GPU times.

Highlights

- Pod detection on low-resolution cocoa tree images
- Comparative study of R-CNN, YOLO and Transformer networks
- True positives, false negatives and false positives
- Carbon footprint assessment
- Cocoa yield assessment

Abstract

Farmers must be able to estimate their crop yields at various growth stages for effective management of their farms and to enable them to interact with cooperatives or traders as early as possible. Here we developed an AI-based cocoa pod detection method using low resolution colour images of cocoa trees on farms in Côte d'Ivoire.

We compared Nano and eXtra-large architectures of six neural networks, including Faster RCNN, Baidu's Real-Time Detection Transformer (RTDetr), DETR-ResNet Vision Transformer, YOLOv5, YOLOv8 and YOLOv11.

These networks were trained with 7,850 annotated cocoa pods on 400 low resolution images, and validated in two independent datasets: a 42 low resolution images containing 990 annotated pods, and a 100 low resolution images containing 2,400 annotated pods.

Unexpectedly, the Nano architectures of the YOLOv8 and YOLOv11 networks outperformed those of the RTDetr networks by ~2% and of the other networks by >5%, with an F1-score of 77% on all images and up to 90% on foreground trees. The dominance of Nano architectures suggests that the eXtra-large architectures, which contain 20-30-times more neurons, may not have been fully trained due to insufficient data, thereby limiting the objectivity of inter-network comparisons. Finally, although the average detection performance of RTDetr for cocoa pods was only 2% lower than that of the YOLO8 network, it was definitively excluded from the candidate models because its per-image processing time was 15–20% higher than that of YOLOv8 and YOLOv11. However, with a performance sensitivity to data of less than 0.5%, YOLOv8 Nano became the best option.

Introduction

Estimating the yield of a crop at its various growth stages is essential for decision making on disease management, harvesting, storage, transport and marketing. Regarding cocoa (*Theobroma cacao*), a novel deep learning approach was recently presented for predicting cocoa yield using a recurrent neural network that combined spatiotemporal climatic data and statistical data on cocoa production in southwestern Nigeria [1]. However, due to a lack of suitable data, this model could not be transposed to Côte d'Ivoire, which alone accounts for >44% of world cocoa production [2], despite the fact that cocoa farming represents a major socioeconomic challenge for this country [3]. Yield estimation in that country is still based on manual counts, which are time-consuming and often hampered by major counting errors. This situation warrants the design of a low-cost, truly operational solution that could be integrated with current practices, while combining real-time identification of cocoa pods at different growth stages in natural environments with a yield estimation model based on pod counts—this would be essential for cocoa research and the cocoa industry. The main objective of this study was to propose a simple reliable AI-based method for image detection and counting of cocoa pods on low-resolution RGB mobile phone images.

In recent years, several studies have described efficient computer vision systems for fruit detection and yield estimation [4]. Fruit detection involves finding instances of fruit in an image. Computer vision systems for fruit detection involve simple pixel segmentation in terms of density or colour or more advanced machine learning methods based on a combination of colour, shape and texture features computed on images sometimes acquired by multiple or multiband sensors. Advanced machine vision systems for yield estimation use deep learning algorithms for object detection: these are being used to an increasing extent in machine vision systems for yield estimation. Among deep learning algorithms, convolutional neural networks (CNNs) have highly efficient conventional detection performance [5]. A growing number of studies have used neural networks for fruit yield estimation [6], and/or fruit detection [7-8]

and/or fruit cultivar identification [9-10], while relying especially on Faster R-CNNs [11-13] or YOLO [14].

Numerous studies have compared diverse neural networks in terms of deep mechanisms of different families [15], as well as intra-family variations [16-17]. Already in 2020, a comparative study of different *in situ* fruit detection networks for apples, mangoes and oranges [18] placed Faster R-CNN in third position for all fruits, with an average accuracy ~2% lower than the best results obtained using an improved Faster R-CNN, and in second position for mango fruit detection, with a difference of 0.2% compared to the best results obtained with YOLOv3. More recently, a study in which a CNN model was compared to YOLO [19] for tomato detection revealed significantly superior YOLO results. A comparison between a detection transformer and YOLOv8 [20] for orange and sweet orange detection showed that the performance of these two models was relatively equivalent.

With regard to cocoa trees, most of the published studies we found concerned the detection of cocoa pod diseases [21-22] or ripeness [23]. These studies were based on images of cocoa pods, not of cocoa trees. The few studies involving plant pathology detection on cocoa images compared different convolutional neural networks and transformers, with scores ranging from 80 to 90% [24]. [25] compared the two detection networks, i.e. U-Net and a fully convolutional network (FCN), where the latter performed much better, with a score of ~94%, i.e. nearly 2% higher than U-Net. Otherwise, [26] carried out a comparative study of different versions of Faster R-CNN and YOLOv5 detection networks, with the YOLOv5X, Faster R50FPN3x and R101C43x architectures obtaining almost identical scores of 95%.

Here we presented a comparative study of different neural network architectures for pod detection on colour images of cocoa trees. In particular, we compared Faster-RCNN [27], YOLOv5 [28], YOLOv8 [29], YOLOv11 [30], DETR-ResNet Vision Transformer [31] and the Real-Time Detection Transformer [32]. These networks were selected for three main reasons: (1) Faster RCNN is the original architecture that has historically been implemented for fruit detection, (2) the YOLO family has benefitted from continuous advances [33], and (3)

transformer-based models are emerging as successors to CNNs [34]. The simplest and most complex architectures were deployed and the F1-score of each network was compared to assess the extent of precision loss, particularly in cases where the cocoa pod detector could be directly embedded in the smartphone or tablet acquiring the images. CNN ResNet50 and ResNet101 architectures [35] were compared with YOLO's specific architectures. In addition, the complexity of objectively estimating the digital processing carbon footprint [36] was addressed to generate all elements needed to assess the efficiency of future operational solution. This exploratory study was part of a broader context focusing on early yield estimation. The yield estimate for a cocoa plot is often obtained by extrapolating the production estimate for a few cocoa trees chosen at random from the plot; the production of each of these cocoa trees is itself estimated based solely on the cocoa pods visible and counted in a photograph of the tree. The networks should therefore regularly process datasets from different cocoa tree plots. This study did not seek to investigate the optimal performance of each neural network trained and tested on a specific data set; it focused specifically on each trained network (performance) sensitivity to datasets. The study deliberately did not focus on the results of tests carried out during the network training sessions; rather, it focused on the results obtained on different validation datasets that were similar and representative of the data given to networks during early plot yield estimates. The study also aimed to assess which models for detecting cocoa pods could be installed on smartphones in order to eventually offer field solutions.

The first part of this article focuses on the data and their origins, the characteristics of the studied networks and the principles applied for estimating the carbon footprint. The second part presents the values of the indicators used and the main results of the comparative study. Finally, the third part contains a general discussion on the results, their relevance and future prospects.

Materials and methods

Material

Cocoa (*Theobroma cacao*) is a small evergreen tree of the Sterculiaceae family. It produces edible cocoa beans with different flavours depending on the cocoa tree variety, and cocoa is the main ingredient in chocolate making [37]. These beans develop in the fruit (pods) of the cocoa tree. Cocoa pods are elongated and resemble a fairly rounded cucumber. They measure 15-25 cm in length and 6-15 cm in diameter, while weighing 300-500 g depending on the variety. These pods are usually reddish-yellow in colour when ripe (see Fig. 1). Because of the large mass of seeds contained in cocoa pods, the pod surface is covered with numerous small bumps, but also marked by around 10 relatively deep longitudinal grooves.

Figure 1: Forastero cocoa orchards, Abidjan district in western-central Côte d'Ivoire.

Cocoa growing is not mechanised in Côte d'Ivoire [38] and cocoa farms are fairly dense 'natural' orchards (see Fig. 1). Forastero is the most widely grown variety, which accounts for 79% of cocoa production in the country, whereas the Trinitario variety accounts for 20%. Criollo is the rarest cocoa variety, and is currently the least cultivated, accounting for just 1% of all harvested cocoa [39]. Our study focused exclusively on Forastero cocoa.

Image acquisition

The cocoa image dataset was acquired manually by an operator in cocoa crop fields in Côte d'Ivoire (July 2019). The images were captured using the main camera of a mobile Samsung Galaxy S10 SM-G973F phone in uncontrolled field conditions. No specific protocols were followed for image acquisition, and parameters such as shooting distances and focal lengths were not specified in advance. Consequently, images in the dataset exhibited variations in depth of field and shooting angles. The lighting conditions and fruit occlusion were not controlled, and the images were taken without flash. Images of different random trees were captured, and the distance between the trees and the camera sensor was not consistently recorded. This lack of a predefined image acquisition protocol and the use of a handheld device in real-field conditions accounted for variability in the dataset for some factors such as illumination and object scale. The resulting dataset consisted of 500 colour images with 4,600 x 3,456 pixel resolution. These images were then deliberately resized to a low resolution of 1,008 x 756 pixels in order to reduce processing times on both computer servers and smartphones.

Image annotation

Expert image annotation is very crucial as it enables the creation of datasets to train neural networks, and of validation datasets to assess their performance. This is a tedious and time-consuming task that often requires substantial resources, as the annotation quantity needs to be as high as possible to be able to conduct highly efficient performance studies. The authors deployed the cocoa-fruit-counting project on Zooniverse [40], the world's largest participatory research platform [41]. Initially developed for the study of constellation images, the platform allows millions of volunteers around the world to be called upon to enhance research project resources, which was particularly helpful here for the image annotation stages.

Figure 2 : Zooniverse annotation. The area of interest outlined by the red polygonal line indicated the tree in the foreground. All the cocoa pods visible in the image were outlined by ellipses shown in blue: their adjustment was based on translations, rotations and resizing of a canonical ellipse.

The Zooniverse cocoa pod annotation process consisted of circumscribing each cocoa pod with an ellipse (in blue in Figure 2). As this research was part of a yield estimation study, each Zooniverse operator (trained expert) was asked to outline the foreground tree on each image with a polygon (in red in Fig. 2). Once completed, the annotations were validated and corrected where necessary via the ImageJ platform [42].

Training and validation datasets

The annotated data was divided into disjoint training and validation datasets that did not share any data in common. The labelled annotations were deduced from the Zooniverse ellipses and consisted of rectangular bounding boxes with dimensions ranging from 10 to 100 pixels per side. These bounding boxes were drawn to as accurately as possible enclose the visible portions of the pods on the images. The training dataset included ~7,850 annotations

representing diverse visual cocoa pod features in terms of shape, colour, sunlight conditions or occlusions on >400 images (S400) from different plots: additional dataset has been added to the training dataset to estimate the network's performance during the various training loops. Two distinct un breakable validation datasets were created with the remaining annotated data to assess variability in the trained networks with the aim of ensuring that the training and validation datasets would be representative of the overall data distribution. The first validation dataset (S42) consisted of 42 images that included 990 annotated pods. The second validation dataset (S100) contained 100 images with a total of 2,400 annotated pods. Each validation dataset consisted of images from a single plot, following the recommended data acquisition protocol for early yield estimation. The difference in dataset size is solely due to the difference in plot size. The plots were selected from the same geographical region, with the same cultivation practices and the same varieties of cocoa trees. These two sets were therefore considered similar, the idea being to assess the sensitivity of the detection networks to the data. Furthermore, the polygons outlining the foreground tree were used to generate binary interest masks. These masks were black-and-white images on which the region defined by the polygonal boundary of the area of interest cocoa tree was rendered in white, while the remaining area was black. Added to datasets S42 and S100 respectively, they defined datasets S42WM and S100WM (see Fig. 3).

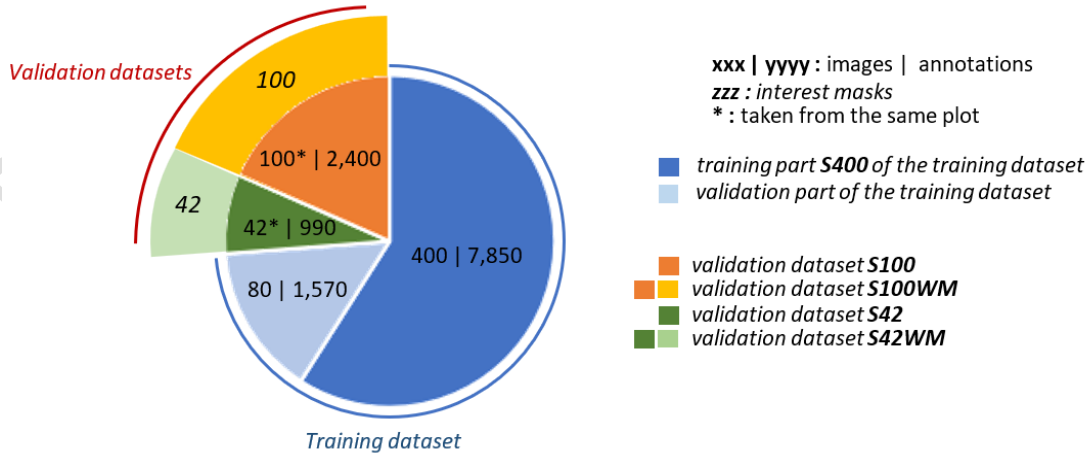

Figure 3: training and validation datasets. The annotated images are divided into three non-overlapping datasets: the S400 training dataset, consisting of 400 images from different plots, and the S42 and S100 validation datasets, consisting of 42 and 100 images from the same plots, respectively.

Neuronal networks

The study was focused on Faster-R-CNN, YOLOv5, YOLOv8, YOLOv11, DETR-ResNet Vision Transformer and Baidu's Real-Time-Detector networks. While the Region-based Convolutional Network (R-CNN), You Only Look Once (YOLO) and Transformers are object detection networks, their respective mechanisms differ. R-CNNs start by finding interesting parts of the image, then examine these parts more closely to determine what they contain. YOLO networks grid the space via successive convolutions and then locate and identify objects of interest. This model first emerged in 2015, but it was not until YOLOv5 (2021) was designed that the model became easily accessible and sufficiently optimised for acceptable detection. YOLOv8 (2023) was then the first model that offered truly significant features and improvements for enhanced detection performance, flexibility and efficiency. YOLOv11 (2024) is the latest iteration in the “classic” series of real-time object detectors. This version includes major improvements in the architecture and learning methods, so it is more versatile than its predecessors. Self-attentive Transformers models (2017) are conceptually based on the attention mechanism and take the context of the observations to be predicted into account. This type of model is particularly effective in translation tasks, where a sequence of words in one language is transformed into a sequence of different words in another language. Vision Transformers emerged in late 2020 and adapted the architecture of Transformers to process image content. These models are rapidly overtaking and replacing convolutional models for computer vision. The Real-Time Detection Transformer features significant optimisations for dealing with multi-scale features—this involved decoupling intra-scale interaction and inter-scale fusion, and making the model highly adaptable, thereby enabling flexible inference speed adjustment.

Yet beyond their differences, these networks share common features. They are all based on a neural architecture i.e., an arrangement of successive layers containing unitary computing elements called neurons. The neurons are all weighted and biased during training in order to be suitable for the detection of objects of interest. The total number of parameters is a linear

combination of the weights, biases, inputs and outputs of each network layer. The Faster-R-CNN Nano ResNet50 architecture (FRCNNN) has around 45 million parameters, the Faster-R-CNN eXtra-large ResNet101 architecture (FRCNNX) around 55 million, the YOLOv5 Nano architecture (YOLOv5N) around 1.8 million, the YOLOv5 eXtra-large architecture (YOLOv5X) around 86 million, the YOLOv8 Nano architecture (YOLOv8N) around 3 million, the YOLOv8 eXtra-large architecture (YOLOv8X) around 86 million, the YOLOv11 Nano architecture (YOLOv11N) around 2.5 million, the YOLOv11 eXtra-large architecture (YOLOv11X) around 57 million, the DETR-ResNet Vision-Transformer Nano ResNet50 architecture (ResNetDetrN) around 26 million, the DETR-ResNet Vision-Tranformer eXtra-large ResNet101 architecture (ResNetDetrX) around 86 million, the Baidu's Real-Time-Detector Large architecture (RTDetrL) around 32 million and the Baidu's Real-Time-Detector eXtra-large architecture (RTDetrX) around 66 million. Furthermore, the networks can only structurally process images of fairly low resolution, i.e. generally between 300 and 1,096 pixels. When images are larger than the resolution limits accepted by the network, the latter automatically resizes the native image, although this sometimes alters the image content.

Image tiling

Small objects of interest may disappear or become indistinguishable after resizing the native images. To avoid this major inconvenience, it is advisable to divide the native image into thumbnails, or so-called tiles, of a resolution that can be processed by the networks, with small objects of interest remaining visible and detectable. A tiling strategy with tile overlap was adopted to ensure that each object of interest would be fully contained within at least one tile. The detection networks therefore provided predictions, i.e. bounding boxes locating cocoa pods in each tile. Post-processing was applied to transfer the cocoa pod detections obtained in each tile to the original image. This involved repositioning the predicted bounding boxes from the individual tiles on the coordinate system of the full image. Due to tile overlapping, duplicate detections of the same cocoa pod could occur across adjacent tiles, so a process was implemented to delete duplicate predictions.

Carbon footprint

A computing server typically comprises two distinct but complementary resources: the service resource that manages the web application interfacing the user and the computations, and the compute resource alternately mobilising the computer's central processing unit (CPU) cores and the graphical processing unit (GPU) cores of the computer graphics card. The GPU manages the storage of temporary data generated during computations. The calculation server is also often associated with a server for storing input data and results produced for periods of varying length, as generally specified in the calculation server guidelines. The whole system is housed in a data centre equipped with power supply and cooling systems.

Estimating the carbon footprint may be very complicated. Various factors must be taken into account, including the energy consumption of the hardware components (CPU, GPU, memory, storage) during the different operational phases (training, validation, inference), the efficiency of the power supply and cooling systems in the data centre, and the source of the electricity powering the infrastructure. While detailed carbon footprint analysis is beyond the scope of this study, information on the CPU and GPU processing times and the temporary data volumes were provided as carbon footprint indicators.

Design of experiments (DOE)

Each network was trained on the S400 training dataset and its performance was evaluated on the two disjoint validation datasets S42 and S100, with or without interest masks. The validation of datasets with interest masks consisted of pairing only expert annotations and network predictions whose geometric centre of the bounding box belonged to the area of interest. Benchmark performances were provided based on the S400 training set. The hyperparameters for each network were left at the default values proposed by their respective authors, who defined the optimal settings for their model. The batch size, which defines the number of tiles processed simultaneously by the network, was arbitrarily set at 32 based on the memory of the graphics card used. The metric chosen during training was the F1-score for convenience: it is a well-known indicator used by the agronomy community. The same

minimalist data enrichment was applied to each network: it consisted solely of systematic vertical inversion of the images on the one hand, and systematic 90° rotations on the other, to 'show' the network of horizontal dents what happens when images accidentally switch to landscape mode due to the orientation of smartphones but are processed in portrait mode. The maximum number of epochs i.e., the maximum number of times the network would see all the data, was set at 1,200 with a patience of 100 epochs, indicating the number of epochs to wait without improvement in validation metrics before stopping training early. Thus, the effective training of YOLO took place between 850 and 930 epochs depending on the model, unlike RCNN. Performance evaluation was conducted at two levels, i.e. at the image level, where all predictions within the entire image were considered, and at the foreground tree level, where the evaluations were focused on or limited to the masks of interest defined per image. This allowed for a detailed analysis of the detection accuracy, both generally and within specific regions.

The normalized rate of geometric overlap between two bounding boxes was used to measure the total or partial superposition of objects. The intersection on union (IoU) of objects was defined by the ratio of the intersection to the union of the two bounding boxes [43]. This latter indicator was applied for the deletion of cocoa pods predicted several times by the networks due to tile overlap bands and for matching expert annotations with network predictions. The value of this ratio was compared with a pre-determined threshold and used to identify overlapping boxes considered to represent the same cocoa pod. The object detection performance was estimated according to the F1-score [44], which is a standard comparative test between the image truth (expert annotations) and the network predictions. The F1-score was the harmonic mean between the precision and recall, i.e. two statistical indicators giving the contribution of 'false positives' and 'false negatives' respectively to the network's overall detection error: true-positives were cocoa pods annotated by the expert and correctly detected by the network, false-negatives were cocoa pods annotated by the expert but not detected by

306 the network, and false-positives were also cocoa pods detected by the network but not
307 considered as such by the expert.

308 The performance indicators were assessed by cocoa pod size class. Based on the dimensions
309 of the ground truth bounding boxes, cocoa pods framed by boxes smaller than 16x16 pixels
310 were considered ‘small’, cocoa pods framed by boxes between 16x16 and 32x32 pixels were
311 considered ‘medium’, and cocoa pods that did not fit into either of the above categories were
312 considered ‘large’.

313 All the experiments were carried out on the AgroDeep platform [45], thereby ensuring that the
314 technical environments were relatively similar between the different tests. Python scripts for
315 pre-processing the data, post-processing the results and inferring the neural networks were
316 encapsulated in Singularity containers [46], which provided an optimal operating environment
317 (operating system, additional calculation packages, CUDA drivers, etc.) for each network in
318 the study. All training and validation procedures were carried out on the same computing
319 server, which was exclusively dedicated to the study, and equipped with a 24GB Nvidia Quadro
320 RTX6000 graphics card, 4,600 CUDA cores and 570 Tensor cores.

321 The tile size was set at 640x640 for training all the networks and at 640x640 with a 200 pixel
322 overlap for validating all the networks. The threshold for removing multiple predictions was set
323 at 0.25. A minimum IoU threshold of 0.7 was set to determine a successful match between the
324 network predictions and the expert ground truth annotations.

325 Results

326 The S400, S42 and S100 datasets illustrated the network performances at the entire image
327 scale, while the S42WM and S100WM datasets illustrated these performances solely at the
328 foreground tree scale. Unsurprisingly, the most recent networks (published as of 2023) gave
329 better results than the older networks (see Fig. 4). The top three were clearly RTDetr, YOLOv8
330 and YOLOv11. With an F1-score of ~77% for S42 and S100, the Nano architecture of
331 YOLOv11 gave markedly better results than YOLOv8 and RTDetr, with F1-scores ranging from

75.3% to 75.7%. The trend was, however, reversed for the S42WM and S100WM datasets, i.e. with an F1-score of ~90%, the YOLOv8 Nano architecture was around 2% better than YOLOv11 and RTDetr. With a maximum difference of 3%, the F1-scores of the Nano and eXtra-large architectures of the YOLOv8 and YOLOv11 networks were very close, or even equivalent, while being substantially higher than those of the RCNN, YOLOv5 and transformers, with a difference of >5%, all architectures combined. Pre-experimentation on the validation of the different networks trained on a set of 255 images containing 4,527 annotated cocoa pods led to a systematic performance decrease of 5-15% on the S42, S100, S42WM and S100WM datasets.

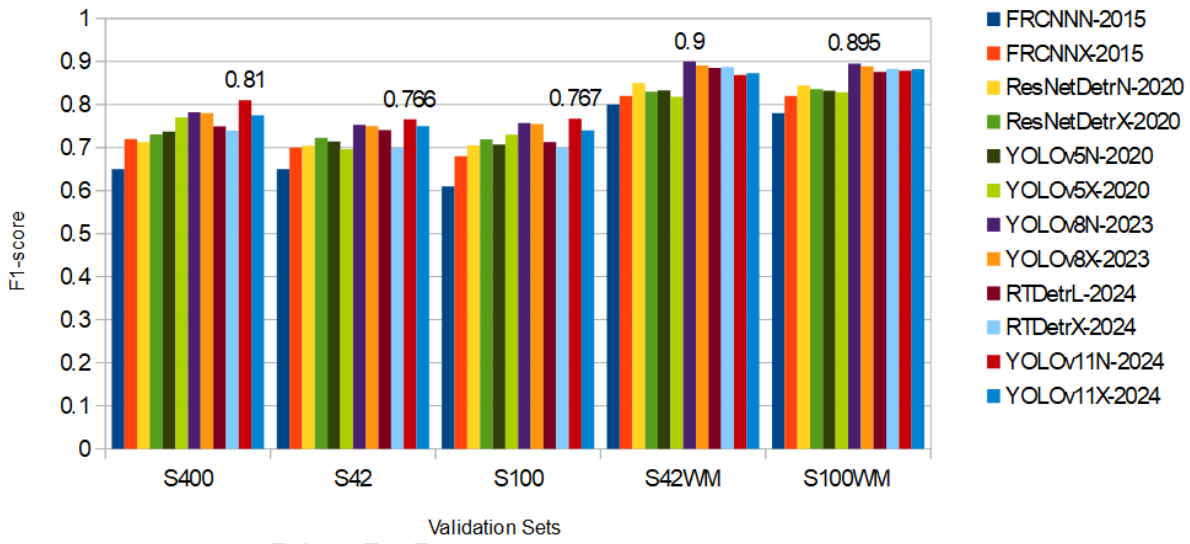

Figure 4: summary of the performance of different networks. Performance was measured by the F1 score of each network trained under similar conditions on each of the validation sets: the S400 set provided the reference performance of the networks on their own training set.

The histograms for S42 and S100 (resp. S42 WM and S100WM) were similar in shape and value, thus indicating the relative stability of the response of these networks when assessed with the different validation datasets.

The F1-scores obtained for S400 were 2-4% higher than those obtained for S42 and S100, as expected since the S400 dataset was exclusively composed of the network training images. This slight drop in performance observed for validation datasets containing data that the

networks had not encountered during their training highlighted the model's generalisation potential.

Assessing the sensitivity of network performance to data sets was difficult with so little data available. However, trends were already emerging when comparing network performance obtained on the S42 and S100 validation datasets, first without and then with the use of interest masks (see Fig. 4). The largest variation observed was 4% for Faster RCNN; this variation fell below 1% for Detr-ResNet transformer, YOLOv5 and the large-RTDetr transformers, and fell to less than 0.5% for the others. The candidate networks were therefore those that performed close to 90% with a variation of less than or equal to 0.5%.

We therefore focused our study on the top three networks with high performance on validation datasets and low performance variations between validation datasets.

The F1-scores of the Nano and eXtra-large architectures of the YOLOv8, YOLOv11 and RTDetr networks by cocoa pod size class (see Tab.1) were used to refine the above overall results, with the lowest scores indicated in red, the medium scores in yellow and the highest scores in green. The lowest scores were obtained for small probably embryonic cocoa pods in the foreground or mature fruit in the background, while the highest scores were obtained for large probably mature cocoa pods on foreground trees. This initial result confirmed that the networks had more difficulty in detecting small background fruit than large foreground fruit.

Table 1 : F1-scores in percentage of the Nano and eXtra-large architectures of the YOLOv8, YOLOv11 and RTDetr networks by cocoa pod size class. Colour coding of the cells ranging from red for the lowest scores to green for the highest scores highlighted the networks' poor ability to detect small cocoa pods in the background and their strong ability to detect large pods in the foreground.

	YOLOv8N			YOLOv8X			YOLOv11N			YOLOv11X			RTDetrL			RTDetrX		
size classes →	*small*	*med.*	*large*	*small*	*med.*	*large*	*small*	*med.*	*large*	*small*	*med.*	*large*	*small*	*med.*	*large*	*small*	*med.*	*large*
S400	46.97	77.08	92.54	50.04	77.78	92.73	50.98	82.16	92.97	43.95	78.68	91.58	45.23	72.32	91.3	46.13	73.51	93.46
S42	31.75	85.62	94.92	31.6	85.8	95.1	33.91	80.98	93.26	35.73	84.6	91	37.63	81.05	91.47	35.48	81.99	93.33
S100	45.51	83.5	95.05	49.06	83.25	94.5	48.58	83.65	94.9	39.3	83.76	93.31	43	77.71	92.05	43.65	80.46	94.22
S42WM	54.55	89.05	95.31	55.74	88.62	93.59	52.17	85.2	93.63	46.67	87.94	91.78	25	86.19	94.37	37.21	86.96	95.2
S100WM	45.71	88.44	95.69	47.9	88.12	95.36	42.38	86.34	95.44	41.33	88.46	94.09	30.93	85.3	94.1	41.1	87.2	94.1
av. NWM	41.41	82.07	94.17	43.57	82.28	94.11	44.49	82.26	93.71	39.66	82.35	91.96	41.95	77.03	91.61	41.75	78.65	93.67
av. WM	50.13	88.74	95.5	51.82	88.37	94.48	47.28	85.77	94.53	44	88.2	92.94	27.96	85.74	94.23	39.15	87.08	94.65

The penultimate row of the table gives the average scores per fruit size class for cocoa pod detection on whole images, while the last row gives the cocoa pod detection scores only in terms of the trees of interest. A slight improvement in the average F1-scores per network was observed when the cocoa pod detections were limited to foreground areas on the images of the trees of interest.

The networks could be ranked according to their respective performances solely based the analysis of the results obtained for the trees of interest. An average F1-score of 43% for small cocoa pod detection, but with a dispersion of ~9%, showed that the RTDetr architectures did not perform well in this size range. An F1-score of 87% for medium-sized pod detection with a dispersion of <1.5% illustrated the similarity of the responses of the three networks, although YOLOv8 had a slight advantage over YOLOv11 and RTDetr. With an F1-score of 94% for large cocoa pod detection with a dispersion of <1%, YOLOv8, YOLOv11 and RTDetr confirmed their ability to identify and count pods on foreground trees. With average F1-scores of 78% across all size classes, the eXtra-large and Nano architectures of the YOLOv8 network were ~3% better than YOLOv11 and ~6% better than RTDetr. But with an F1-score of 95.5% for the large fruit class alone, the YOLOv8 Nano architecture performed >1% better than all the other architectures.

A qualitative assessment was achieved by images showing the match between expert annotations and network predictions: the true-positives were represented by blue bounding boxes, the false-negatives by yellow bounding boxes and the false-positives by pink bounding boxes in the zones of interest on the foreground tree. Although not easy to appraise, the blue and pink boxes showed the location of the network predictions, which meant that there could potentially have been slight differences between networks.

On the foreground tree image in Figure 5, Nano YOLOv8 (c) and eXtra-large RTDetr (g) obtained the fewest errors, with 22 true-positives and two false-positives, for a local F1-score of 95.6%. All annotated cocoa pods were detected, and also the network detected two false-positives: the pink box in the centre of the image was a FALSE false-positive, i.e. a real pod

not annotated by the expert; the pink box furthest to the left was apparently related to a background cocoa pod captured by the network due to an insufficiently tight contour around the foreground tree.

Figure 5: Qualitative visual assessment of prediction accuracy for YOLOv11 (top), YOLOv8 (middle) and RTDetr (bottom) in Nano (left) and eXtra-large (right) architectures. The prediction boxes are coloured according to the annotation/prediction appearing classes:

Paradoxically, eXtra-large YOLOv8 (d) also missed a cocoa pod, despite having a more complex architecture than Nano YOLOv8. The eXtra-large YOLOv11 architecture (b) seemed to be more efficient than the Nano YOLOv11 architecture (a), which generated numerous detection errors, although not as efficient as the YOLOv8 architecture. With three false-positives and a local F1-score of 93.3%, Nano RTDetr (f) ranked numerically *ex aequo* with networks presenting one false-negative and two false-positives (b,d), even the absence of false-negatives was a good thing.

CPU and GPU times

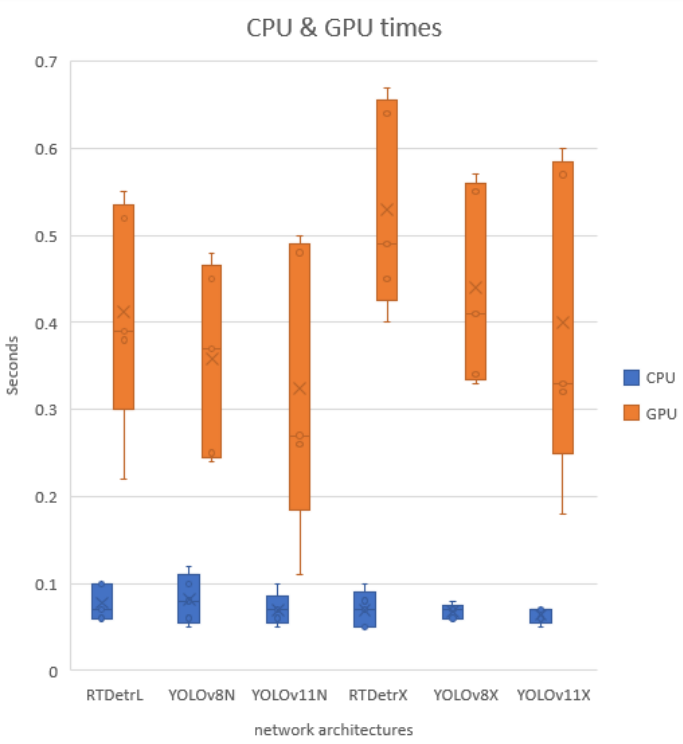

Figure 6: Mean CPU and GPU times for RTDetr, YOLO8 and YOLO11 validations per image. Average times were assessed solely on the S400, S100 and S42 validation datasets.

The CPU and GPU runtimes for training and network validation were defined by differences between the system times taken at the start and end of each key stage. The server used was exclusively dedicated to the calculations so as to ensure inter-network comparability of calculation times. Input data tiling, removal of multiple predictions, pairing of expert annotations and network predictions and processing performance estimations were carried out via the computer's CPU, whereas the learning and prediction of networks were carried out via the

computer's GPU. The CPU time to load data from the hard disk to the graphics card memory during GPU processing was overlooked.

The average unit times per type of computing resource, evaluated on the validation of the five batches S400, S100, S42, S100WM and S42WM, accounted for the resources consumed per network. The boxplots (see Fig. 6) summarise the datasets at six key figures, i.e. the minimum (lower horizontal line), the 25th percentile (lower circle), the median (horizontal line), the mean (cross), the 75th percentile (upper circle) and the maximum values (upper horizontal line), thereby facilitating the detection of outliers in the distribution of resource consumption values, which could indicate instances of particularly high or low resource usage. With a tiny dispersion of 0.0066 s, the average CPU times/image between the different networks were considered to be similar—a very logical result given that the input data pre-processing and output post-processing algorithms were rigorously identical. With a significant dispersion of 0.071 s, the GPU times illustrated the impact of the different complexities and mechanisms of the architectures tested. RTDetr transformer was clearly the slowest network, with an average GPU time/image of 0.41 s for its simplest architecture and 0.53 for its most complex architecture. YOLOv8 was the fastest network, with an average time of 0.32 s for its simplest architecture and 0.40 s for its most complex architecture. However, with the exception of the Nano architecture of the YOLOv8 network, the boxplots showed outlier distributions, with the median value tending towards the 25th percentile. For YOLOv8, the median tended towards the mean.

Discussion and prospects

With overall F1-scores <80% on all images, and barely 90% on foreground trees, cocoa pod detection was less successful than the detection of apples and sweet peppers [11], mangoes [12], oranges [14] and lemons [20], which had F1-scores of 92-98%. The score estimation depended on the validation dataset contents, which could likely be explained by the nature and repeatability of the training data. The number of annotations (or annotated images)

seemed very sufficient, yet the use of Zooniverse induced major bias. Since the same images were successively annotated by several users, the datasets included multiple annotations of the same cocoa pods, although there were no real differences in context. The 11,240 annotations involved in the study accounted for <2,500 truly different cocoa pod contexts. There were not enough examples to constitute sufficiently representative training and validation datasets. The relationship between the training dataset size and the network performance has long been known [47], as was confirmed here by network performance differences of 5-15% when comparing training with a set of 225 annotated images and with a set of 400 annotated images.

Assessing the foreground tree on low-resolution images sometimes proved difficult, as the viewpoint could not be changed. Drawing a single polygon to outline the foreground tree proved restrictive, i.e. cocoa pods of background trees were visible in the polygon of interest circumscribing the foreground tree due to the low cocoa tree foliage density; the same problem also arose for cocoa pods on peripheral foreground tree branches located within the polygon outlining the tree of interest. However, when detected by the network, these specific pods were considered false positives and contributed to increasing the network's error rate. This observation highlighted a major difference between the approaches taken by the expert and the neural network: the expert mentally identified the branches of the tree of interest on which he had to annotate the cocoa pods, unlike the neural networks, which detected all the pods in the area of interest, regardless of the tree on which they were growing. The accuracy of the delineation of the area of interest was therefore crucial.

The defined acquisition protocol allowed shooting distances in the 3-12 m range to be fully in line with field operators' practices and constraints. A tree photographed at 3 m distance thus showed relatively large cocoa pods, whereas the same pods appeared small when photographed at 12 m distance. This flexible shooting range should enhance the detection network robustness and enable operators to implement a size class approach. Our cocoa pod size class assessment method partitioned the data and substantially improved the results, i.e.

boosting the F1 scores to ~94%, yet it could be even further improved by switching from pixel to metric dimensions. When using centimetric dimensions, the images were calibrated from reference patterns on the foreground tree or from image metadata as described by [47]. The cocoa pod detection networks were ranked according to two complementary criteria: the value of their respective performance and the low sensitivity to the data of this performance. This performance was assessed using the F1 score, widely used in thematic publications [48]; it had the dual advantage of being widely used and understood by agronomists in their own validation studies while incorporating detection excesses (false positives) and defects (false negatives). Direct interpretation of the inter-network classification results was therefore complex. Score differences >5% were considered relevant enough to validate the inter-network ranking, but differences <5% were not considered sufficiently relevant: we considered that these differences could be explained more by the datasets used than by the actual network capabilities. In this specific case, the performance of the networks could be considered equivalent, and the final choice of model to be implemented in an operational solution could be based on an additional criterion such as processing time. Beyond this ranking process, our study aimed to assess whether the drop in performance of the Nano architectures which, unlike the eXtra-large architectures, favoured speed over precision [30], was still acceptable for performance estimates. Against all expectations, the YOLOv8 and YOLOv11 Nano architectures achieved F1-scores that were a few hundredths higher than those of the eXtra-large YOLO and RTDetr architectures. Figure 5 clearly illustrated the difficulty of the eXtra-large YOLOv8 and YOLOv11 architectures in detecting all pods on the foreground tree, thereby suggesting that these networks had not been properly trained. The eXtra-large architectures had 20- to 30-times more parameters to weight than the Nano architectures. Indeed, they would have needed much more data to achieve training of equivalent quality [49], which would be necessary to be able to truly objectively compare the networks.

The performance sensitivity study was difficult to conclude due to the obvious lack of data. A dozen additional validation datasets from different cocoa plots would have been necessary;

however, two validation datasets were sufficient to identify the main trends and a few rules for selecting candidate neural networks. It is not unreasonable to exclude from the list those networks whose performance variations exceeded 0.5%.

The CPU times per image were stable, even though inter-network variations of <0.007 s were observed. These variations did not markedly influence the time required to count cocoa pods in a plot yield estimation workflow. CPU times were measured on the basis of the 5 validation datasets and were artificially transformed into an average time per image, while the processing of foreground trees alone included an additional post-procession script to exclude expert annotations and network detections outside of the polygons circumscribing the foreground trees. Otherwise, a marked 0.007 s fluctuation in GPU time was noted. This could undoubtedly be explained by the different complexities of the assessed neural architectures, e.g. the YOLOv11 Nano architecture had 2 million parameters, as compared to the YOLOv8 eXtra-large architecture with 68 million parameters. GPU times for cocoa pod detection would therefore have an impact on the time to count cocoa pods in a plot yield estimation workflow. Yet the GPU time seemed to be an interesting criterion that could potentially be used to complement the network classification. RTDetr networks were discarded because their average GPU times per image were 15-20% higher than those of YOLOv8 and YOLOv11, even though their average cocoa pod detection capacity was only 2% lower than that of YOLOv8.

The persistent and temporary data storage sizes were easy to evaluate for the different neural architectures when the algorithmic implementations were perfectly mastered. For example, the weight file size was constant for a given neural architecture, while ranging from a few kilobytes to several megabytes depending on the case. Systematic data tiling for training and validation enabled *a priori* evaluation of the temporary data size, while precise specification of the treatment processes enabled evaluation of the persistent data size. Here we deliberately have not elaborated on these aspects since it was not possible to obtain a realistic assessment of the carbon footprint due to the lack of precision in estimating the CPU and GPU times.

However, the trends that emerged from this initial research should help us draw up guidelines for future studies.

It would thus be essential to enrich the datasets with truly new annotations representative of the different field contexts. This would significantly improve the performance and robustness of the network, while increasing the amount of data to confirm the results drawn from this study and enable more meaningful statistical studies on processing times to be conducted. A calibration pattern, e.g. a blue sphere of known diameter, could be introduced to enable image calibration, which in turn would enhance the precision on cocoa pod distributions by size class. The trends noted in this first comparative study will now have to be confirmed in further research, particularly regarding the predominance of the YOLOv8 Nano architecture. If these trends are confirmed, the real performance of this latter architecture will have to be assessed with an embedded system so as to obtain all of the elements needed to define the technical foundations of an operational solution to help estimate the yield of cocoa tree plots.

Conclusion

In this study, the lightest and heaviest architectures of the Faster R-CNN, Detr-ResNet Vision Transformer, Baidu's Real Time Detection Transformer, YOLOv5, YOLOv8, and YOLOv11, with identical hyperparameters, were trained with 7,850 annotated pods from 400 low-resolution images of different cocoa plots. Each network architecture was validated on two complementary datasets that had not been involved in the training phases. Two validation datasets were defined in accordance with the recommended data acquisition protocol for early yield estimation, each corresponding to a cocoa plot. The S42 validation dataset comprised 42 images containing 990 annotated cocoa pods, and the S100 dataset comprised 100 images containing a total of 2,400 annotated cocoa pods.The validations were based on two modalities: detection of cocoa pods in the entire set of images, and otherwise in the foreground tree circumscribed by a specific polygonal line.

Unsurprisingly, the most recent networks published as of 2023 gave better results than the older networks. The top three networks were clearly RTDetr, YOLOv8 and YOLOv11. With average F1 scores close to 48% for detecting cocoa pods in the background, 88% for those in the middle, and 94.5% for those in the foreground, all with a maximum dispersion of 2%, the study confirmed that the YOLOv8, YOLOv11 and RTDetr networks were good candidates for identifying and counting cocoa pods on foreground trees with sufficient accuracy. However, with an F1 score of ~77% for the detection of cocoa pods in all images and 90% for foreground cocoa trees, the YOLOv8 Nano architecture outperformed the other two by almost 3%.

Beyond this ranking, our study was designed to assess whether the drop in the performance of Nano architectures—which, unlike the eXtra-large architectures, prioritized speed to the detriment of precision—was still acceptable for yield estimates. Against all expectations, the YOLOv8 and YOLOv11 Nano architectures achieved F1-scores that were a few hundredths higher than those of the eXtra-large YOLO and RTDetr architectures. Consequently, the eXtra-large architectures—with 20- to 30-times more parameters to adjust than the Nano architectures—would probably have required larger datasets to be able to set up truly relevant network training sessions so as to enable fully objective inter-network comparisons.

However, one of the major aspects for cocoa tree plot yield estimation was the performance sensitivity to data from different networks. Two validation sets did not provide certainty, only trends. With a sensitivity of less than 0.5%, YOLOv8 Nano took the lead in the top three.

With a dispersion of <0.007 s, the average CPU times per image between the different networks were considered to be similar, i.e. this slight fluctuation clearly had no impact on the cocoa pod detection workflow. With a dispersion of 0.07 s, the average GPU times per image reflected the impact of variations in the complexity of the architectures of the different networks studied. The GPU time was a complementary network classification indicator to choose the best trained network to implement in the cocoa pod counting workflow. With average GPU times per image 15-20% higher than those of the YOLOv8 and YOLOv11 networks, the RTDetr Large and eXtra-large architectures were discarded even though their average cocoa pod

detection capacity was just 2% lower than that of the Nano YOLOv8. As the persistent and temporary data storage sizes were easy to evaluate according to the different neural architectures, the trends that emerged from our preliminary study should help establish guidelines for future studies to define a low-carbon footprint tool for early yield estimation in cocoa plantations.

Acknowledgements

This study was conducted within the framework of the Cocoa4Future (C4F) project, which is funded by the European DeSIRA Initiative under grant agreement No. FOOD/2019/412-132 and by the French Development Agency. L-CB was funded by Phenome project (ANR-11-INBS-0012). Juan-Pablo Rojas-Bustos is acknowledged for his participation in earlier discussions related to this work.

The authors would also like to thank David Manley for his proofreading and rewriting work.

To Crunchy, my 13-year-old extra-dwarf rabbit who died peacefully in my arms while I was writing the first version of this article.

References

1. Olofintuyi S. S., Olajubu E. A., Olanike D. An ensemble deep learning approach for predicting cocoa yield. Heliyon. 2023;9(4).
2. Kongor J. E., Owusu M., Oduro-Yeboah C. Cocoa production in the 2020s: Challenges and solutions. CABI Agriculture and Bioscience. 2024;5(1), 102.
3. Alain B. K. Economic impact of cocoa culture in Ivory Coast and in Ghana from 1980 to 2015. J His Arch & Anthropol Sci. 2024;*9*(2), 62-67.
4. Gongal A., Amatya S., Karkee M., Zhang Q., Lewis, K. Sensors and systems for fruit detection and localization: A review. Computers and electronics in agriculture. 2015;116, 8-19.
5. Girshick R., Donahue J., Darrell T., Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. Proceedings of the IEEE conference on computer vision and pattern recognition. 2014;580–587.
6. Hou L., Wu Q., Sun Q., Yang H., Li P. Fruit recognition based on convolution neural network. Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD). 2016; 12th International Conference on, (IEEE), 18-22.
7. Rahnemoonfar M., Sheppard C. Deep count: fruit counting based on deep simulated learning. Sensors. 2017;17(4), 905.
8. Xiao F., Wang H., Xu Y., Zhang R. Fruit detection and recognition based on deep learning for automatic harvesting: An overview and review. Agronomy. 2023;13(6), 1625.
9. Borianne P., Sarron J., Borne F., Faye E. Deep mango cultivars: cultivar detection by classification method with maximum misidentification rate estimation. Precision Agriculture. 2023;24(4), 1619-1637.
10. Daviet B., Fournier C., Cabrera-Bosquet L., Simonneau T., Cafier M., Romieu C. Ripening dynamics revisited: an automated method to track the development of asynchronous berries on time-lapse images. Plant Methods. 2023;19(1), 146.

11. Sa I., Ge Z., Dayoub F., Upcroft B., Perez T., McCool C. Deepfruits: A fruit detection system using deep neural networks. Sensors. 2016;16(8), 1222.
12. Borianne P., Borne F., Sarron J., Faye É. Deep Mangoes: from fruit detection to cultivar identification in colour images of mango trees. arXiv preprint. 2019;1909.10939.
13. Wan S., Goudos, S. Faster R-CNN for multi-class fruit detection using a robotic vision system. Computer Networks. 2020;168, 107036.
14. Mirhaji H., Soleymani M., Asakereh A., Mehdizadeh S. A. Fruit detection and load estimation of an orange orchard using the YOLO models through simple approaches in different imaging and illumination conditions. Computers and Electronics in Agriculture. 2021;191, 106533.
15. Cheng R. A survey: Comparison between Convolutional Neural Network and YOLO in image identification. In Journal of Physics: Conference Series. 2020; Vol. 1453, No. 1, p. 012139). IOP Publishing.
16. Sharma A., Kumar V., Longchamps L. Comparative performance of YOLOv8, YOLO9, YOLOv10, YOLOv11 and Faster R-CNN models for detection of multiple weed species. Smart Agricultural Technology. 2024;9, 100648.
17. Khanam R., Asghar T., Hussain M. Comparative Performance Evaluation of YOLOv5, YOLOv8, and YOLOv11 for Solar Panel Defect Detection. In Solar, MDPI. 2025; Vol. 5, No. 1, p. 6.
18. Wan S., Goudos S. Faster R-CNN for multi-class fruit detection using a robotic vision system. Computer Networks. 2020;168, 107036.
19. Raj R., Nagaraj S. S., Ritesh S., Thusha, T. A., Aparanji V. M. Fruit classification comparison based on cnn and yolo. In IOP Conference Series: Materials Science and Engineering. 2021; Vol. 1187, No. 1, p. 012031.
20. Jrondi Z., Moussaid A., Hadi M. Y. Exploring End-to-End object detection with transformers versus YOLOv8 for enhanced citrus fruit detection within trees. Systems and Soft Computing. 2024; 6, 200103.

21. Mamadou D., Kacoutchy J. A., Ballo A. B., Kouassi B. M. Cocoa pods diseases detection by MobileNet confluence and classification algorithms. International Journal of Advanced Computer Science and Applications. 2023;14(9).
22. Vera D. B., Oviedo B., Casanova W. C., Zambrano-Vega C. Deep learning-based computational model for disease identification in cocoa pods (Theobroma cacao L.). arXiv preprint arXiv: 2024;2401.01247.
23. Ayubi A., Faiz M., Situmorang G. B., Ramadhani K. N., Utama N. P. A Cocoa Ripeness Detection and Classification Model Based on Improved YOLOv5s. In 2023 10th International Conference on Advanced Informatics: Concept, Theory and Application (ICAICTA). 2023;1-6. IEEE.
24. Sykes J. R., Denby K. J., Franks D. W. Computer vision for plant pathology: A review with examples from cocoa agriculture. Applications in Plant Sciences. 2024;12(2), e11559.
25. Ayikpa K.J., Mamadou D., Sodjinou S.G., Ballo A.B., Gouton P., Adou K.J. Detecting and Extracting Cocoa Pods in the Natural Environment Using Deep Learning Methods. In International Conference on Digital Technologies and Applications, Springer Nature Switzerland. 2023;164-174
26. Lammoglia S. K. D., Borianne P., Théveny F., Cabrera-Bosquet L. Real-time Image detection of cocoa pods in natural environment using deep learning algorithms. ICCO. 2023.
27. Ren S., He K., Girshick R., Sun J. Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems. 2025; 28, 91-99
28. Liu Y., Lu B., Peng J., Zhang Z. Research on the use of YOLOv5 object detection algorithm in mask wearing recognition. World Sci. Res. J. 2020; 6(11), 276-284.
29. Reis D., Kupec J., Hong J., Daoudi A. Real-time flying object detection with YOLOv8. arXiv preprint arXiv. 2023;:2305.09972.

30. Khanam R., Hussain M. YOLOv11: An overview of the key architectural enhancements. arXiv 2024. arXiv preprint arXiv:2024;2410.17725.
31. Carion N., Massa F., Synnaeve G., Usunier N., Kirillov A., Zagoruyko, S. End-to-end object detection with transformers. In European conference on computer vision 2020:(pp. 213-229). Cham: Springer International Publishing.
32. Zhao Y., Lv W., Xu S., Wei J., Wang G., Dang Q., ... Chen J. Detrs beat yolos on real-time object detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2024;16965-16974.
33. Jegham N., Koh C.Y., Abdelatti M., Hendawi A. Evaluating the evolution of yolo (you only look once) models: A comprehensive benchmark study of yolo11 and its predecessors. arXiv preprint arXiv. 2024;2411.00201.
34. Arkin E., Yadikar N., Xu X., Aysa A., Ubul, K. A survey: object detection methods from CNN to transformer. Multimedia Tools and Applications. 2023;82(14), 21353-21383.
35. He K., Zhang X., Ren S., Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2016;770-778.
36. Benaben D., Berthoud F., Guennebaud G., Ligozat A.L., Valcke S. Estimation de l'empreinte carbone d'une heure de calcul sur un cœur CPU ou sur un GPU (Doctoral dissertation, Labos 1point5). 2024.
37. Oddoye E.O., Agyente-Badu C.K., Gyedu-Akoto E. Cocoa and its by-products: Identification and utilization. Chocolate in health and nutrition. 2013;23-37.
38. Sabas B.Y.S., Danmo K.G., Madeleine K.A.T., Bogaert J. Cocoa production and forest dynamics in ivory coast from 1985 to 2019. Land, 2020;9(12).
39. Dago M. R., Zo-Bi I. C., Konan I. K., Kouassi A. K., Guei S., Jagoret P., Hérault B. What motivates West African cocoa farmers to value trees? Taking the 4 W approach to the heart of the field. People and Nature. 2025;7(1), 215-230.
40. Lammoglia S.K.D., Cabrera-Bosquet L., Rojas-Bustos J.P. 2022. https://www.zooniverse.org/projects/phenoarch/cocoa-fruit-counting

41. Simpson R., Page K. R., De Roure D. Zooniverse: observing the world's largest citizen science platform. In Proceedings of the 23rd international conference on world wide web. 2014;1049-1054.
42. Schneider C.A., Rasband W.S., Eliceiri K.W. NIH Image to ImageJ: 25 years of image analysis. Nat Methods. 2012;9, 671–675.
43. Rahman M.A., Wang Y. Optimizing intersection-over-union in deep neural networks for image segmentation. In International symposium on visual computing. 2016;234-244. Cham: Springer International Publishing.
44. Yacouby R., Axman D. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In Proceedings of the first workshop on evaluation and comparison of NLP systems. 2020;79-91..
45. Borianne P., Théveny F., Bertrand B., Villain L., Faye É., Sarron J., Viennois G., Borne F., Jaeger J. L'IA au service de l'agriculture : au cœur de l'expérience Agro'Deep. 2021. ⟨hal-05019239⟩
46. Kurtzer G.M., Sochat V., Bauer M.W. Singularity: Scientific containers for mobility of compute. PloS one, 2017;12(5), e0177459.
47. Bailly A., Blanc C., Francis É., Guillotin T., Jamal F., Wakim B., Roy P. Effects of dataset size and interactions on the prediction performance of logistic regression and deep learning models. Computer Methods and Programs in Biomedicine. 2022;213, 106504.
48. Vasconez J.P., Delpiano J., Vougioukas S., Cheein F.A. Comparison of convolutional neural networks in fruit detection and counting: A comprehensive evaluation. Computers and Electronics in Agriculture. 2020;173, 105348.
49. Dawson H.L., Dubrule O., John C. M. Impact of dataset size and convolutional neural network architecture on transfer learning for carbonate rock classification. Computers & Geosciences. 2023;171, 105284.