Vol. 03, Iss. 2 (2026) 106-114, DOI: 10.61552/JSI.2026.02.005

JSDI

Journal of Sustainable Development Innovations

www.jsi.aspur.rs

Mobile-Friendly-Based Cocoa Leaves Disease Detection Deep Learning Model Using YOLOv11

Suale Yakubu[a,*], Agnes Mindila[b], Peter Kihato[c]

[a]Pan African University Institute for Basic Sciences, Technology and Innovation, Department of Electrical Engineering, Nairobi, Kenya,
[b]Jomo Kenyatta University of Agriculture and Technology, Department of Computing and Information Technology, Nairobi, Kenya,
[c]Jomo Kenyatta University of Agriculture and Technology, Department of Electrical and Electronic Engineering, Nairobi, Kenya.

Keywords:

Cocoa disease detection, YOLOv11, Mobile-friendly deep learning model, Precision agriculture, Artificial intelligence

* Corresponding author:

Suale Yakubu
E-mail:
yakubu.suale@students.jkuat.ac.ke

Received: 7 December 2025
Revised: 2 February 2026
Accepted: 4 April 2026

A B S T R A C T

Uncovering knowledge about rural areas is a significant challenge for the effective use of modern technological diagnostic tools, such as disease detection and diagnostic systems. It is essential to understand the extent of cocoa diseases to ensure the stability of farm productivity across Africa, especially in West Africa. This paper presents a mobile-friendly machine learning model to detect and classify anthracnose, black pod disease (BPD), cocoa swollen shoot disease (CSSD), vascular streak disease (VSD), and healthy leaflets. The proposed solution addresses challenges of accessibility to cocoa disease detection and classification machine learning models to assist cocoa farmers in improving productivity. A lightweight YOLOv11 model was developed, compatible with edge computing devices, which is easily accessible to cocoa farmers across Africa to assist farmers in detecting various known cocoa diseases affecting crop yield. The results obtained reveal a remarkable performance with a mAP@50 greater than 98% for the majority of classes and an average accuracy and recall exceeding 95%. The model has been successfully converted to TensorFlow Lite for easy deployment on mobile devices. This approach allows for automated, rapid, and accessible diagnosis for farmers, helping to improve crop health and food security.

1. INTRODUCTION

In West Africa, the impact of cocoa and cocoa products is a vital economic activity, assisting a large number of smallholder farmers, particularly in Côte d'Ivoire and Ghana [1, 2]. These two countries contribute approximately 60% or more of the global cocoa production [3, 4]. However, cocoa plants and trees are vulnerable to diseases such as anthracnose and cocoa swollen shoot virus disease (CSSVD), threatening the cocoa yields, agricultural economy, and cocoa farmers' income at large. In the past years, the cocoa framers largely depended on agricultural officers in the disease identification process during visual inspection, which is costly, slow, and inaccessible to farmers in remote areas[2, 5, 6]. However, advancements in deep learning (DL) and machine learning (ML) present a highly recommended opportunity in revolutionizing the cocoa disease detection in various cocoa farms[7-10]. The You Only Look Once (YOLO) algorithm is known for its impact in computer vision and object detection [11, 12]. YOLOv11 was introduced in the YOLO Vision 2024 conference (YV24) as one of the latest YOLO vision series, which is an advanced step towards real-time object detection. YOLOv11 adopts a redesigned architecture that provides better feature extraction, increased accuracy, and improved processing speed, while maintaining a compact model size [13]. It stands out for its performance on various computer vision tasks such as object detection, classification, segmentation, pose estimation, and oriented detection [14]. Its high efficiency and versatility make it an ideal model for real-time vision applications. This paper employed YOLOv11 to explore the development of a mobile-friendly deep machine learning (DML) model to accurately identify multiple diseases on images of cocoa leaves. The solution aims to optimize the model for edge devices such as Android and smartphones used by cocoa farmers to ensure usability and accessibility across all cocoa farmers in West Africa. Moreover, by leveraging a dataset with annotated images from agricultural institutions across Africa, the study aims to build a robust mobile-friendly AI-driven application for detecting cocoa diseases in real-time to enhance early intervention strategies, reduce the dependence on agricultural officers in the identification of cocoa disease, and mitigate crop losses. This proposed solution ultimately empowers cocoa farmers in strengthening resilience against emerging cocoa diseases in cocoa production and safeguarding livelihoods.

2. RELATED WORK

Recent development and application of machine learning have gained attention in digital agriculture, significantly impacting disease detection and diagnoses in cocoa production industries. This provides farmers with an efficient and accessible solution for easily identifying multiple diseases from an agricultural perspective. This section presents a review of existing work on machine learning and digital agriculture mechanisms in the identification of disease. Several researchers explore the application of machine learning in agriculture, for instance [7] explored the implications and importance of digital agriculture using the state-of-the-art technology for small-scale farmers in Colombia, emphasizing the potential of technology-driven solutions in enhancing productivity and decision-making. It highlights the ways digital tools can bridge knowledge gaps and improve crop management, underscoring the relevance of machine learning incorporation into agricultural practices. Similarly, [8] worked on a DL-based computational model capable of cocoa pod disease detection utilizing convolutional neural networks (CNNs) for cocoa pods analysis and classification of images with diseases, demonstrating the efficacy of a machine learning approach in plant pathology and setting precedents for application in cocoa pod detection. Moreover, [9] proposed a smartphone application that employs machine learning for the identification of cocoa disease, leveraging CNNs for processing images captured in real-time disease diagnostic feedback. The work highlights the implications and impact of a mobile-friendly artificial intelligence (AI) solution that is accessible to smallholder cocoa farmers. Other researchers [10, 15] presented a hybrid combination of CNN and XGBoost classifier for cocoa leaf disease identification, integrating a machine learning approach with feature extraction to enhance classification generalization and accuracy. The research contributes to the improvement of disease recognition abilities in multiple disease identification on the same leaf. These researchers collectively demonstrated the potential of digital agriculture and DL and DML in disease identification and diagnosis. Despite the fact that several researchers have done their best in this scope, there is a need for deploying a mobile-friendly AI-driven solution for low-resource devices or edge computing devices to assist farmers in easily detecting diseases, ensuring accessibility to cocoa farmers globally

3. PROPOSED MODEL

This section highlights the dataset collection, model training, and predictions to demonstrate the performance of the proposed YOLOv11 model on an edge device as a lightweight version for cocoa leaf disease detection. The proposed model aims to utilize ML models for cocoa disease prediction and classification, with the focus on edge computing devices such as Android smartphones and OIS phones, leveraging the gap of accessibility to a lightweight ML model for disease identification across various edge computing devices.

3.1 Dataset Collection and Annotation

The dataset utilized in this paper was provided by Amini AI, Makere AI Lab, Makeree University Research Innovations Fund (RIF), and Macroni Machine Learning Lab through Zindi Africa [16]. The dataset was presented for the Amini cocoa contamination challenge globally as open-access data for competitors. The challenge was to develop the best mobile-friendly AI model for cocoa disease identification, with a primary focus on identifying anthracnose and cocoa swollen shoot disease (CSSD). This study modified the dataset to include the additional detection of vascular Streak disease (VSD) and Black Pod disease (BPD). The Fig. 1 illustrates the classification made to labels to enhance easy detection and identification of various diseases.

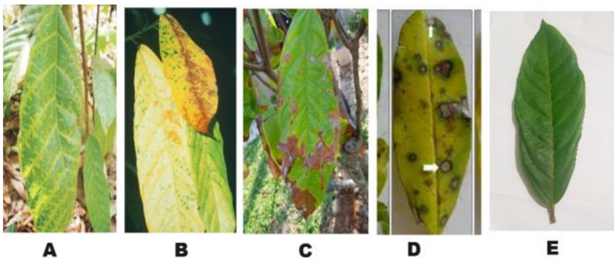

Fig. 1. Example of cocoa leaves with disease classification.

The dataset comprises a total of 5712 images, including healthy leaves and infected leaves with corresponding visible symptoms. The dataset was customized with five classes, namely, anthracnose, CSSD, VSD, BPD, and healthy, respectively, as shown in the Fig. 1. This dataset was labeled and annotated using Label Studio and exported as yolo format for training and prediction accuracy. Table 1shows the distribution of image annotations.

Table 1. Annotated Class Distribution.

Classes	CSSD	VSD	BPD	Anthracnose	Healthy	Total
No. of images	1142	188	209	1902	2271	5712

The images are distributed across the various classes to improve the model's performance. The distribution was done based on the number of available images collected. While the dataset contains a significant number of images, totaling 5,712, there is an imbalance in the class distribution, with Anthracnose and healthy classes dominating, covering approximately 70% of the entire dataset. This may lead to lower accuracy at BPD and VSD. The imbalance was addressed using targeted data augmentation techniques to ensure that the training set of underrepresented classes is enriched, thereby maintaining balanced model detection performance across different types of disease [17].

3.2 Overview of proposed YOLOv11

YOLO11 is one of the latest models in the Ultralytics YOLO series for real-time object detection, built upon previous versions of YOLO while redefining higher cutting-edge accuracy, efficiency, and speed [18]. YOLO11 architecture has become a versatile choice for computer vision activities due to its improved architecture and training methods [19]. Fig. 2 illustrates the YOLO11 architecture utilized in this paper.

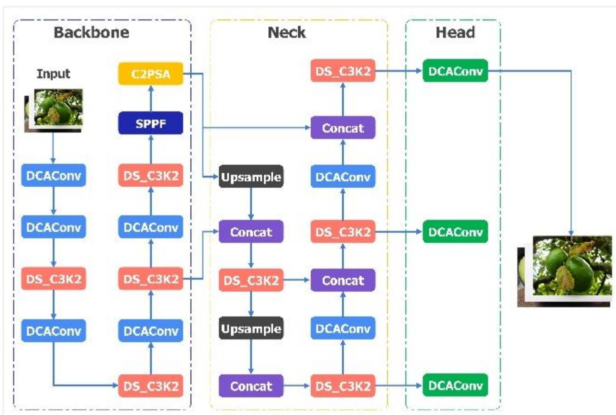

Fig. 2. Yolo11 Network Architecture.

The core of the YOLO architecture is built on three fundamental components, of which the first is the backbone, serving as the basic feature extractor. It is empowered by convolutional neural networks that transform raw images into multi-scale feature maps.

The second component is the neck, which serves as an intermediate data processing layer to aggregate and improve the representation of features across different scales. The third component is the head, functioning as the prediction mechanism in generating the final results for object classification and localization in line with the featured maps [20].

3.3 Model training and predictions

The proposed mobile-friendly model was trained using the dataset obtained from Amini AI and a custom dataset to evaluate the effectiveness of the model. The general flow of the implementation of the proposed mobile-friendly model is illustrated in Fig. 3.

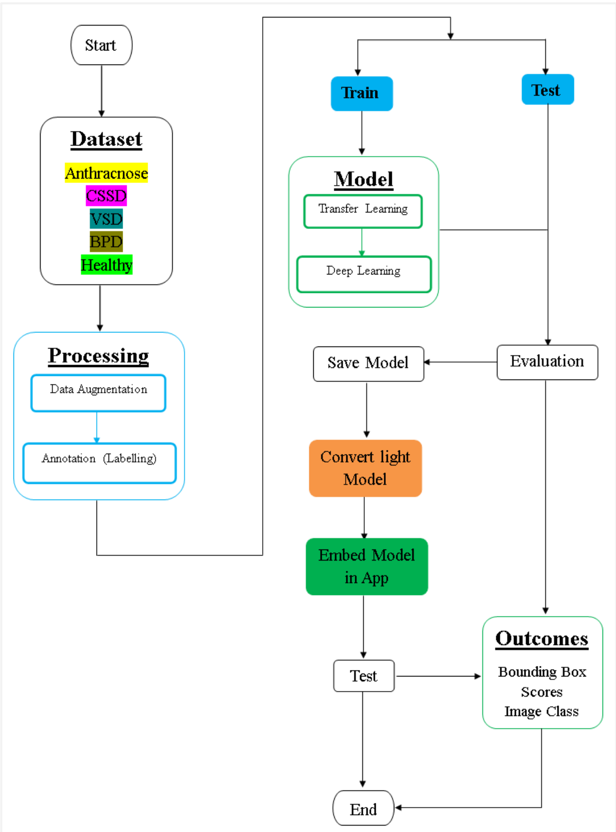

Fig. 3. Proposed Model Flowchart.

Maintaining a proper dataset is vital, as it directly impacts the model's performance due to the inclusion of diverse and high-quality images. Once a proper dataset is obtained, it undergoes data augmentation and annotations to improve the capabilities of generalizing the new images, serving as the foundation of the model training. The next phase after data augmentation and annotation is the proposed AI model training using YOLO11 algorithms as the base model, which is recognized for its high performance in object detection and recognition [21, 22]. The DL process creates room for the model to extract features and patterns from the dataset to enhance the accurate classification of new images [23-25].

To maintain the standard input dimensions of the dataset for training the yolo algorithm and to improve generalization, an image resize ratio of 640x640 pixels was applied. A robust augmentation pipeline was applied during the training of the dataset, including 90^0 rotations, 2.5-pixel Gaussian blur, 15% grayscale, horizontal flipping, and 1.5% random noise injection. This was to simulate real-world variations in the cocoa leaf orientation, background conditions, and lighting to enhance the model's accuracy in diverse field environments.

The dataset was split into training (70%), validation (20%), and testing (10%) subsets to ensure robust model evaluation [26]. The following models, yolov8, yolov9, and yolov10, were also trained using the Adam optimizer with a learning rate of 0.005 and a batch size of 64 to ensure stable convergence and conduct a comparative experiment with the proposed yolov11 to validate its effectiveness on edge computing devices.

3.4 Evaluations

After the model was successfully trained, evaluated, and saved, it was converted to a lightweight version optimized for mobile deployment using TensorFlow Lite. The evaluation performance metrics include precision, accuracy, recall, and F1-score, which ensure the reliability of the proposed model.

The formula 1 to 4 illustrates the mathematical models of the respective performance metrics.

$$P_r = \frac{T_p}{T_p + F_p} \tag{1}$$

$$F_1 = 2 \times \frac{P_r \times R_c}{P_r + R_c} \quad (2)$$

$$A_{cc} = \frac{T_p + T_n}{T_p + T_n + F_p + F_n} \quad (3)$$

$$R_c = \frac{T_p}{T_p + F_n} \quad (4)$$

Were; P_r = precision, F_1 = F1-Scores, T_p = true positive, T_n= true negative, F_p= false positive, F_n = false negative, A_{cc} = accuracy, and R_c = recall.

3.5 Machine Specifications

The workstation used for training the model features an Intel processor, 9th generation, 16GB of DDR4 RAM, a 512GB NVMe SSD, and a 4GB NVIDIA Quadro GPU with Windows 11 installed, which supports deep learning model training, fast data access, and efficiently handling large datasets. The specifications for resource-constrained devices for testing and evaluating performance are Android 8.0+ and iOS 13+ with a minimum RAM of 2GB and ARM64 architecture.

3.6 Model Export and Conversion

To ensure the trained YOLO model is compatible for offline mobile integration and usage, the following conversion method was applied to convert the model to a lite version.

1. Install the Microsoft C++ build tool compatible with compiling TensorFlow Lite (TFlite) libraries.
2. Install supported libraries: tf-keras, sng4onnx>=1.0, onnx_graphsurgeon>=0.3.26, "onnx2tf>1.17.5,<=1.26.3", protobuf>=5, and tflite-support to allow the conversion of best.pt into TFlite format for easy integration into the mobile app development environment.
3. Use a Python script to convert the best.tp to TFlite using the snapshot in the Fig. 4.

```
from ultralytics import YOLO

# Load your trained YOLOv11 model
model = YOLO(r"C:\Users\mrsmile\best.pt")  # Path to your .pt file

# Export directly to TFLite
model.export(format="tflite")

print("☑ Model successfully converted to TFLite!")
```

Fig. 4. Snapshot for converting PyTorch to TFLite.

4. Save the converted model as a .tflite file for mobile development integration.

3.7 Mobile App deployment

The lightweight models were then integrated into a mobile-compatible application to further evaluate the performance of the proposed model in a real-world, mobile-friendly application domain. This involves testing the mobile application with a variety of images under different environmental and lighting conditions to validate the model's consistency in detecting and classifying diseases. The deployment process is illustrated in Figure 5.

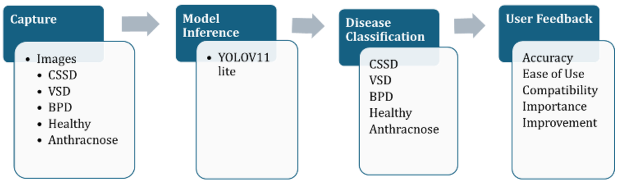

Fig. 5. End-to-End Mobile Disease Detection Job Flow.

User tests the model on their Android or iOS phones by opening the lightweight app to capture cocoa leaf images, which are then analyzed and classified into the disease based on the classes used in this project by the YOLOv11 lite model. The detection results contain a bounding box, image class, and scores serving as a determination of the accuracy of the model in cocoa disease prediction on an edge device. This process ensures the deployment of an efficient mobile-friendly ML model for cocoa leaf disease predictions, enabling farmers to make informed decisions and take timely preventive measures to ensure better cocoa yield. The mobile application(app) was built and deployed on Flutter to support Android devices running a minimum version 5.0 and above, including the latest Android version 14.0. It is also compatible with iOS 13 plus to ensure broader compatibility with the latest Android and iOS smartphones for easy accessibility to farmers for utilization.

4. RESULTS AND DISCUSSION

This section presents the outcomes of the model based on precise performance metrics, indicating the ability of its efficient detection and classification of various cocoa diseases.

4.1 Proposed Model results

The proposed model was evaluated against several standard object detection metrics, including accuracy (P), recall (R), the average of average accuracies at 50% of IoU (mAP@50), and the overall average of mAPs of 0.5 to 0.95 (mAP@50–95). The Table 2 summarizes the model's performance on the five classes studied: Anthracnose, BPD, CSSVD, VSD, and Healthy Leaves.

Table 2. Class with various performances.

Classes	Precision (P)	Rappel (R)	mAP@50	mAP@50-95
Anthracnose	1.0	0.96	0.99	0.88
Black Pod	0.99	0.94	0.98	0.85
CSSVD	0.99	0.99	0.99	0.87
Healthy	0.95	0.97	0.99	0.89
VSD	0.99	0.99	0.99	0.85

The results obtained demonstrate very high performance of the model, especially for the Anthracnose class, which achieves a perfect accuracy of 1.00 and an mAP@50 of 0.99. The CSSVD and VSD classes also achieved very high scores, reflecting the model's ability to effectively capture discriminating features despite the visual complexity of some diseases. The mAP@50–95, which is a stricter metric incorporating various intersection over Union (IoU) thresholds, remains above 0.85 for all classes, confirming that the model is not satisfied with a good approximate location but produces accurate predictions.

The normalized confusion matrix illustrated in Fig. 6 presents a minimal rate of inter-class errors as an indicator of a low confusion matrix between visually similar diseases. The normalized confusion matrix indicates the performance of the classification model, which classified above 99% of class images, demonstrating its ability to distinctly identify and classify various diseases on cocoa leaves.

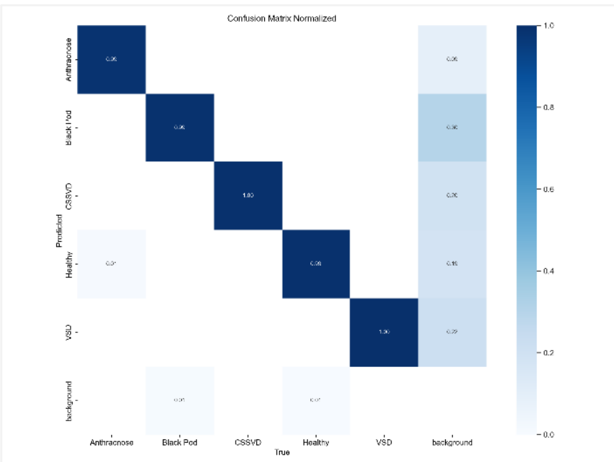

Fig. 6. Normalized confusion Matrix.

The precision-recall curve illustrated in Fig. 7 confirms the balance between sensitivity and specificity, with a high area under the curve for all classes. It demonstrates a high precision of 99.2% across all classes, utilizing the IoU threshold of 0.5, which is the standard performance indicator in object detection and multiclass classifications [27].

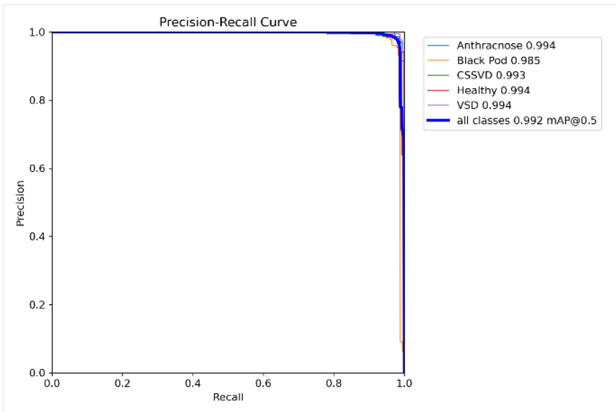

Fig. 7. Precision-Recall curve.

The prediction result, shown in Figure 8, was based on images taken under different environmental conditions, having different classes of diseases, which depicts the model's ability to be efficiently deployed on resource-constrained mobile devices for real-time detection. Each detection includes a bounding box, a class label, and an associated percentage of scores, allowing end users to interpret the results. The proposed model effectively detects 98% of anthracnose disease, 93% of healthy images, 84% of black pod disease, 99% of CSSVD, and 89% of VSD. This model not only performs better but is also optimized for edge devices for easy accessibility and utilization in local communities.

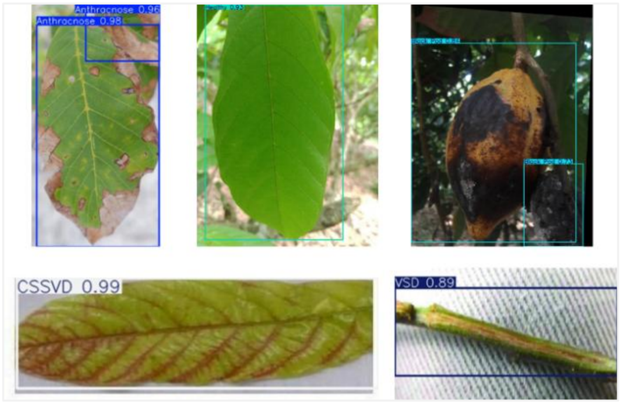

Fig. 8. Prediction Results.

In summary, the lightweight model, optimized with TensorFlow Lite, proved to be suitable for use on smartphones and other devices with low computing power. This paves the way for mobile applications for farmers, especially in rural areas, to quickly identify cocoa diseases and thus strengthen early intervention strategies.

4.2 Comparison with Other Models

To further evaluate and justify the suitability of the proposed YOLOv11n for deployment on mobile devices, a comparative evaluation was made against older YOLO family versions (YOLOv8n, YOLOv9c, YOLOv10n, and YOLOv11n), shown in Table 3. The evaluations capture the key performance metrics, including training completion time, inference time, and computational complexity - Giga Floating Points Per Second (GFLOPs).

Table 3. Comparison of the proposed model with others.

Model	YOLOv8n	YOLOv9c	YOLOv10n	YOLO11n
Precision (P)	0.977	0.985	0.919	0.981
Rappel (R)	0.978	0.982	0.881	0.967
mAP@50	0.992	0.991	0.967	0.991
mAP@50-95	0.893	0.952	0.872	0.87
Inference	9.9ms	133.6ms	12.5ms	10.4ms
Train Time	0.941 hrs	19.053 hrs	1.202 hrs	1.025 hrs
GFLOPs	8.1	102.3	8.2	6.3

The comparative analysis drawn from Table 3 is that YOLOv9c achieved the highest performance metrics in recall of 98.2% and an excellent mAP@50-95 of 95.2%. However, it utilized a higher computation cost of 102.3 GLOPs and a longer training period of 19.05 hours, making it impractical for deployment on resource-constrained and edge-computing devices like smartphones. Similarly, YOLOv8n also demonstrates a higher metric with faster inference speed of 9.9ms and training duration of 0.941 hours, but requires higher computation complexity, making it less recommended for resource-constrained devices [28]. YOLOv10n, on the other hand, demonstrated the lowest metrics with higher inference, training period, and GLOPs, highlighting a limitation in robustness compared to YOLOv11n.

Finally, the proposed YOLOv11n achieved high metrics while maintaining a lightweight computational complexity of 6.3 GFLOPs with a shorter training time of 1.025 hours compared to YOLOv9c and YOLOv10n [29]. It demonstrated a balanced trade-off with faster inference of 10.4ms, making it suitable for mobile deployment without compromising detection accuracy and computing resources [30]. Overall, YOLOv11n has proven to be the most practical model, striking a balance between accuracy, efficiency, and hardware compatibility, making it the preferred architecture for cocoa disease detection in resource-limited environments.

4.3 Limitations and future direction

Despite the great performance metrics of the proposed model, limitations such as leaf orientation and illumination exist in field deployment. This poses a challenge to disease detection accuracy and reliability. In addition, the proposed model does not address the issue of an automatic mechanism for emerging disease strains. Future research should focus on incorporating an automatic update mechanism and a multi-label detection mechanism for detecting emerging disease strains.

5. CONCLUSION

In this paper, a mobile-friendly machine learning model was developed and evaluated on a cocoa leaves' disease dataset. The model was successfully optimized for mobile devices, with a principal focus on compatibility with edge computing devices, accuracy, and efficiency. The Yolov11 model was trained and optimized for an edge device for real-time cocoa disease detection and classification. The results demonstrate an excellent performance in identifying and classifying various common cocoa diseases, while being fast in prediction and lightweight, making it suitable for remote areas' cocoa farmers to utilize. The results show that the proposed model achieved an average accuracy of 99% and an average background noise of 22%, demonstrating its ability to reduce background noise while maintaining accuracy. Also, the model achieved a precision of 99.2% across all classes and an IoU threshold of 0.5, depicting a high performance in a standard object detection and multiclass classification environment. This model can be integrated into mobile applications for instant diagnosis, without dependence on an Internet connection or powerful IT infrastructure. This makes it a promising tool for the fight against cocoa diseases, especially in rural areas where access to phytosanitary experts is limited. However, the model is limited to detecting cocoa disease on an edge device but does not incorporate diagnostic techniques, necessitating future work to explore the integration of explainable AI mechanisms, such as Grad-CAM and SHAP, to guide farmers on a precise preventive mechanism.

REFERENCES

[1] A. Kroeger, H. Bakhtary, F. Haupt, and C. Streck, *Eliminating Deforestation from the Cocoa Supply Chain*, Mar. 2017, doi: 10.1596/26549.

[2] M. Chukwujama, "Corporate reporting on modern slavery in global supply chains: Examining Nigeria's cocoa farms," Oct. 2023, doi: 10.26686/WGTN.24434464.

[3] Zindi Africa, "Amini Cocoa Contamination Challenge," [Online]. Available: https://zindi.africa/competitions/amini-cocoa-contamination-challenge/data.

[4] N. Heiss, J. Meier, U. Gessner, and C. Kuenzer, "A review: Potential of Earth observation (EO) for mapping small-scale agriculture and cropping systems in West Africa," *Land*, vol. 14, no. 1, p. 171, 2025, doi: 10.3390/land14010171.

[5] S. Jones and P. Gibbon, "Developing agricultural markets in Sub-Saharan Africa: Organic cocoa in rural Uganda," *J. Dev. Stud.*, vol. 47, no. 10, pp. 1595–1618, 2011, doi: 10.1080/00220388.2011.579107.

[6] S. Carodenuto, "Governance of zero deforestation cocoa in West Africa: New forms of public–private interaction," *Environ. Policy Gov.*, vol. 29, no. 1, pp. 55–66, 2019, doi: 10.1002/eet.1841.

[7] J. M. Melo Velasco, "Digital agriculture implications for small farmers: Evidence from Colombia," Aug. 2023, doi: 10.32469/10355/96993.

[8] B. Vera, B. Oviedo, W. C. Casanova, and C. Zambrano-Vega, "Deep learning-based computational model for disease identification in cocoa pods (*Theobroma cacao* L.)," 2024, [Online]. Available: http://arxiv.org/abs/2401.01247.

[9] S. Kumi *et al.*, "Cocoa companion: Deep learning-based smartphone application for cocoa disease detection," *Procedia Comput. Sci.*, vol. 203, pp. 87–94, 2022, doi: 10.1016/j.procs.2022.07.013.

[10] K. S. Kouassi *et al.*, "Detection of cocoa leaf diseases using the CNN-based feature extractor and XGBoost classifier," *Open J. Appl. Sci.*, vol. 14, no. 10, pp. 2955–2972, 2024, doi: 10.4236/ojapps.2024.1410193.

[11] M. Hussain, "YOLO-v1 to YOLO-v8, the rise of YOLO and its complementary nature toward digital manufacturing and industrial defect detection," *Machines*, vol. 11, no. 7, p. 677, 2023, doi: 10.3390/machines11070677.

[12] K. Han *et al.*, "A survey on vision transformer," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 87–110, 2023, doi: 10.1109/TPAMI.2022.3152247.

[13] R. Khanam and M. Hussain, "YOLOv11: An overview of the key architectural enhancements," 2024, [Online]. Available: http://arxiv.org/abs/2410.17725.

[14] T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object detection using YOLO: Challenges, architectural successors, datasets and applications," *Multimed. Tools Appl.*, vol. 82, no. 6, pp. 9243–9275, 2023, doi: 10.1007/s11042-022-13644-y.

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE CVPR*, 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[16] Zindi Africa, "Amini Cocoa Contamination Challenge," [Online]. Available: https://zindi.africa/competitions/amini-cocoa-contamination-challenge.

[17] N. Khalal, A. A. E. Djamai, I. Keraghel, and M. Nadif, "Imbalanced data clustering via targeted data augmentation using GMM and LLM," *Lect. Notes Comput. Sci.*, vol. 15669, pp. 246–260, 2025, doi: 10.1007/978-3-031-91398-3_19.

[18] S. Tomassini *et al.*, "The YOLO framework: A comprehensive review of evolution, applications, and benchmarks in object detection," *Computers*, vol. 13, no. 12, p. 336, 2024, doi: 10.3390/computers13120336.

[19] G. Jocher and J. Qiu, "Ultralytics YOLO11," 2024, [Online]. Available: https://github.com/ultralytics/ultralytics.

[20] Z. Tian *et al.*, "An optimized YOLOv11 framework for the efficient multi-category defect detection of concrete surface," *Sensors*, vol. 25, no. 5, p. 1291, 2025, doi: 10.3390/s25051291.

[21] B. Murthy *et al.*, "Investigations of object detection in images/videos using various deep learning techniques and embedded platforms—A comprehensive review," *Appl. Sci.*, vol. 10, no. 9, p. 3280, 2020, doi: 10.3390/app10093280.

[22] C. S. Lin and H. Y. Hsieh, "An automatic defect detection system for synthetic shuttlecocks using transformer model," *IEEE Access*, vol. 10, pp. 37412–37421, 2022, doi: 10.1109/ACCESS.2022.3165224.

[23] L. Alzubaidi *et al.*, "A survey on deep learning tools dealing with data scarcity: Definitions, challenges, solutions, tips, and applications," *J. Big Data*, vol. 10, no. 1, 2023, doi: 10.1186/s40537-023-00727-2.

[24] S. K. Baduge *et al.*, "Artificial intelligence and smart vision for building and construction 4.0: Machine and deep learning methods and applications," 2022, doi: 10.1016/j.autcon.2022.104440.

[25] K. Choudhary *et al.*, "Recent advances and applications of deep learning methods in materials science," 2022, doi: 10.1038/s41524-022-00734-6.

[26] S. Shahrabadi *et al.*, "Automatic optimization of deep learning training through feature-aware-based dataset splitting," *Algorithms*, vol. 17, no. 3, p. 106, 2024, doi: 10.3390/a17030106.

[27] Z. Jia *et al.*, "FedDAD: Solving the islanding problem of SAR image aircraft detection data," *Remote Sens.*, vol. 15, no. 14, p. 3620, 2023, doi: 10.3390/rs15143620.

[28] S. Khatoon *et al.*, "Environmental sustainability through optimal energy consumption using IoT-based edge computing and image processing," 2024, doi: 10.4018/979-8-3693-5748-4.CH013.

[29] Y. Ma *et al.*, "I-YOLOv11n: A lightweight and efficient small target detection framework for UAV aerial images," *Sensors*, vol. 25, no. 15, p. 4857, 2025, doi: 10.3390/s25154857.

[30] R. Kishor, "Performance benchmarking of YOLOv11 variants for real-time delivery vehicle detection: A study on accuracy, speed, and computational trade-offs," *Asian J. Res. Comput. Sci.*, vol. 17, no. 12, pp. 108–122, 2024, doi: 10.9734/AJRCOS/2024/V17I12532.