Journal of
Electronics Technology Exploration

Homepage: htttps://shmpublisher.com/index.php/joetex
p-ISSN 3025-3470 | e-ISSN 3026-1066

Disease Detection in Cocoa Fruit Using YoloV4 Convolutional Neural Network Architecture

Alfaiz Alafi Luthfie[1], Farihah[2]
[1,2] Department of Computer Science, Universitas Negeri Semarang, Indonesia

Article Info

Article history:

Received Jan 20, 2026
Revised Jan 26, 2026
Accepted Feb 8, 2026

Keywords:

CNN
Cocoa fruit
Detection of fruit diseases
YoloV4

ABSTRACT

Detecting diseases in cocoa beans is important to prevent a decline in crop quality and economic losses. This study aims to detect diseases in cocoa beans using the YOLOv4 Convolutional Neural Network (CNN) architecture. The dataset used consists of images of cocoa beans, which then undergo preprocessing, data division, data augmentation, and modeling. The test results show that the YOLOv4 model is capable of detecting diseases in cocoa beans with an accuracy rate of 97%, demonstrating good performance in the classification and object detection process. However, this study still has limitations in terms of the amount of data and the variety of image capture conditions. Therefore, further research is expected to use a larger dataset and test the model in more diverse environmental conditions to improve the reliability of the system.

Corresponding Author:

Alfaiz Alafi Luthfie,
Department of Computer Science,
Universitas Negeri Semarang,
Semarang, Indonesia.
Email: faizalafiluthfie@students.unnes.ac.id
https://doi.org/10.52465/joetex.v3i2.648

1. INTRODUCTION

Cocoa is one of Indonesia's strategic plantation commodities, playing a crucial role in the national economy. Indonesia is one of the world's top three cocoa producers, and this commodity is a source of livelihood for millions of small farmers [1], [2]. However, cocoa productivity still faces various obstacles, one of which is the attacks of pests and diseases that cause a decline in the quality and quantity of crops. One of the main pests that is difficult to control is the Cocoa Fruit Borer (CFB), which attacks the inside of the fruit, damaging the cocoa beans and thereby directly impacting production yields [3].

In the field, farmers still largely identify cocoa diseases manually. They depend on visual observation. This method relies heavily on the farmer's experience and is prone to errors. Different diseases often show similar symptoms. As a result, control measures are inaccurate and ineffective [4]. This situation shows the need for a more objective, rapid, and accurate method of detecting cocoa bean diseases.

Advances in image processing technology and artificial intelligence have created significant opportunities in agriculture, particularly for detecting plant diseases. As future work, research can explore the integration of digital image-based methods with other technologies to further improve the accuracy and

scalability of disease identification [5]. One widely used method is deep learning, particularly Convolutional Neural Networks (CNN), which have a high capacity for extracting visual features and classifying images [6].

However, most previous studies still focus on image classification without directly considering the localization of diseased objects. In addition, some studies still use conventional CNN architectures that require considerable computational resources and are less than optimal for real-time applications. Research specifically addressing disease detection in cocoa beans using the object detection approach is also limited, so further study is needed to fill this research gap.

Object detection is an important approach in image processing because it not only classifies objects but also determines the location of objects in an image through bounding boxes. This method involves two main tasks, namely object classification and localization [7]. In recent years, developments in deep learning have produced various high-performance object detection algorithms, one of which is YOLO (You Only Look Once) [8].

YOLOv4 is an improvement on previous versions of YOLO, designed to increase accuracy while maintaining inference speed. This architecture combines CSPDarknet53 as the backbone, PANet as the neck, and various optimization techniques that make it superior at detecting objects in real time [9]. Compared to other methods, such as R-CNN or SSD, YOLOv4 offers a better balance between speed and accuracy, making it more suitable for application in image-based disease detection systems [10].

Based on these issues, this study aims to develop a disease detection system for cocoa beans using the YOLOv4 architecture. This study utilizes images of cocoa beans to automatically detect and classify disease types. The main contribution of this study is the application of YOLOv4 in detecting cocoa bean diseases accurately, quickly, and efficiently, so that it can be used as a tool to assist in the disease identification process in agriculture and support more accurate decision-making for farmers.

2. METHOD

This research was conducted in stages, starting from data collection, data augmentation, data splitting, image annotation, modeling, and model evaluation.

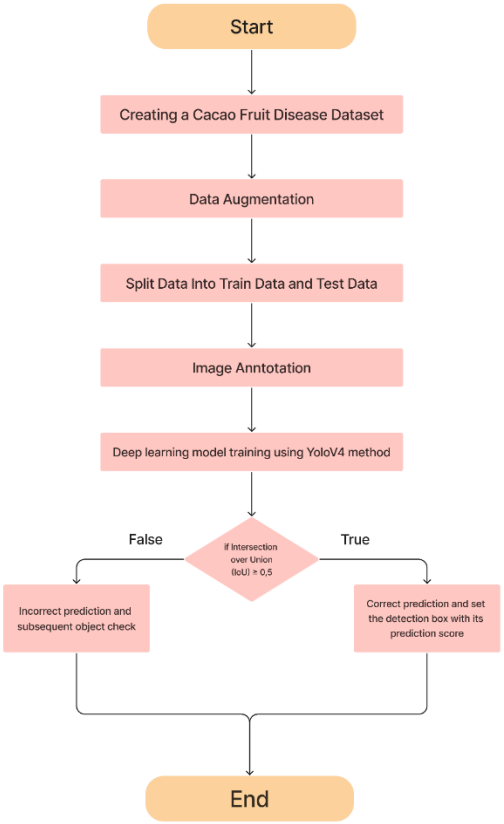


Figure 1. YOLOv4 method flowchart

2.1. Data Collecation

The dataset is taken from the kaggle.com website which contains image data of cocoa fruit which has several classes of diseases such as Fito class which is phytophthora disease, Monilia which is monilia disease

and Sana class which is not affected by disease. The dataset is available at https://www.kaggle.com/datasets/serranosebas/enfermedades-cocoa-yolov4?utm_source=chatgpt.com. From it, this study used 200 images of cocoa beans as experimental data for model training and testing.

2.2. Data Augmentation

The data augmentation procedure was employed in this study to enhance the quality of the data of the datasets that had been collected. Data augmentation is a technique commonly used in deep learning to enrich image variation without altering the original labels, allowing the model to learn more robustly [11]. Augmentation techniques performed on datasets such as horizontal reversal, image rotation by 90, and color transformation (saturation, hue, and exposure). The purpose of performing augmentation is to ensure effective generalization in models for the detection of diseases in cocoa fruits.

2.3. Split Data Into Train Data and Test Data

One important step in creating a machine learning-based system is the separation of datasets into two parts, namely data train and data test. Data train is data used to train machines while test data is used to test the results of training with the aim of ensuring the performance of machine learning. The purpose of dataset separation is to ensure that models that have been trained and built can work well against data that has never been seen before. The dataset of 200 data is divided into 175 data for the data train and 25 data for the test data. This division is performed to ensure that the model can generalize to new data and minimize training bias.

2.4. Image Annotation

Image annotation is the process of adding information into an image according to its class. Image annotation is an important stage in object detection because the quality of the annotation affects the model's ability to recognize objects accurately [12]. Information added in the form of labels, bounding boxes, segmentation, landmarks, and others. The purpose of adding this information is to create a training dataset in machine learning. The information is used for training the model to perform specific tasks such as object detection. The dataset obtained is annotated and numbered using the LabelImg annotation tool application written in Python language.

2.5. Deep Learning Model Training YOLOv4

The classification of cocoa diseases in this study was performed using the YOLOv4 architecture, which is a CNN-based object detection method capable of detecting objects in real time with good accuracy. This method combines various optimization techniques, such as CSPDarknet53 and PANet, to improve object detection performance [9]. Data that has been annotated through the Image Annotation process will be trained with the YOLOv4 architecture. To perform the training process and normalize the dataset requires configuring the YOLOv4 layer to be used as a convolution network. The convolution layer used is 3x3 and features extraction size is 1x1 and max polling layer with kernel size 2x2. These layers are used to form the YOLOv4 network configuration architecture. The result or output of the dataset will be reduced to 26 x 26 x 21, where 26 x 26 is the grid size and 21 is the sum of the filters. For the complete structure can be seen in Figure 2.

```
conv     32       3 x 3/ 1    416 x 416 x   3 ->  416 x 416 x  32 0.299 BF
   1 conv     64       3 x 3/ 2    416 x 416 x  32 ->  208 x 208 x  64 1.595 BF
   2 conv     64       1 x 1/ 1    208 x 208 x  64 ->  208 x 208 x  64 0.354 BF
   3 route  1                                     ->  208 x 208 x  64
   4 conv     64       1 x 1/ 1    208 x 208 x  64 ->  208 x 208 x  64 0.354 BF
   5 conv     32       1 x 1/ 1    208 x 208 x  64 ->  208 x 208 x  32 0.177 BF
   6 conv     64       3 x 3/ 1    208 x 208 x  32 ->  208 x 208 x  64 1.595 BF
   7 Shortcut Layer: 4,  wt = 0, wn = 0, outputs: 208 x 208 x  64 0.003 BF
   8 conv     64       1 x 1/ 1    208 x 208 x  64 ->  208 x 208 x  64 0.354 BF
   9 route  8 2                                   ->  208 x 208 x 128
  10 conv     64       1 x 1/ 1    208 x 208 x 128 ->  208 x 208 x  64 0.709 BF
  11 conv    128       3 x 3/ 2    208 x 208 x  64 ->  104 x 104 x 128 1.595 BF
  12 conv     64       1 x 1/ 1    104 x 104 x 128 ->  104 x 104 x  64 0.177 BF
  13 route  11                                    ->  104 x 104 x 128
  14 conv     64       1 x 1/ 1    104 x 104 x 128 ->  104 x 104 x  64 0.177 BF
  15 conv     64       1 x 1/ 1    104 x 104 x  64 ->  104 x 104 x  64 0.089 BF
  16 conv     64       3 x 3/ 1    104 x 104 x  64 ->  104 x 104 x  64 0.797 BF
  17 Shortcut Layer: 14,  wt = 0, wn = 0, outputs: 104 x 104 x  64 0.001 BF
```

Figure 2. YOLOv4 convolution layer structure

In this training process, the total number of convolution layers used is 36 layers. Each convolutional layer used is worth 21 layers obtained from the formula:
Filter = (number of classes in dataset + 5) x 3..
max_batches = (number of classes) * 2000
For more details can be seen in Table 1.

Table 1. YOLOv4 layer structure

Layer	Kernel Size
Convolution Layer	3 x 3
Features Extraction	1 x 1
Max Polling Layer	2 x 2
Output Dataset	26 x 26 x 21

The data training process is carried out with the help of Google Colaboratory or Google Colab because there is a free GPU with a capacity of 12 GB so that the data training process is faster than devices that have less supporting specifications. Google colab is a free cloud service provided by the Google company in the form of executable documents that are used to edit, write, save and share source code programs that have been written on Google Drive. During the data training process, if the Intersection over Union is more than equal to 0.5 then the system will declare the prediction correct and set the prediction box and prediction score. If the Intersection over Union is less than 0.5 then the system assumes its prediction is wrong and will look for the next object. An example of the training process can be seen in Figure 3.

```
 2114: 1.299200, 1.260810 avg loss, 0.001000 rate, 6.963851 seconds, 135296 images, 27.467754 hours left
Loaded: 19.957913 seconds - performance bottleneck on CPU or Disk HDD/SSD
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 139 Avg (IOU: 0.878555), count: 12, class_loss = 0.020300, iou_loss = 39.023964, 
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 150 Avg (IOU: 0.897894), count: 27, class_loss = 0.036456, iou_loss = 18.719423, 
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 161 Avg (IOU: 0.875471), count: 10, class_loss = 0.001923, iou_loss = 1.267904, t
 total_bbox = 940286, rewritten_bbox = 0.075509 %
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 139 Avg (IOU: 0.812828), count: 20, class_loss = 0.529806, iou_loss = 73.092354, 
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 150 Avg (IOU: 0.881399), count: 27, class_loss = 0.243015, iou_loss = 20.502394, 
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 161 Avg (IOU: 0.919123), count: 16, class_loss = 0.024772, iou_loss = 2.228604, t
 total_bbox = 940349, rewritten_bbox = 0.075504 %
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 139 Avg (IOU: 0.852211), count: 46, class_loss = 0.653393, iou_loss = 135.183105,
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 150 Avg (IOU: 0.883854), count: 52, class_loss = 0.550761, iou_loss = 35.377228, 
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 161 Avg (IOU: 0.881571), count: 14, class_loss = 0.031108, iou_loss = 1.427243, t
 total_bbox = 940461, rewritten_bbox = 0.075495 %
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 139 Avg (IOU: 0.758239), count: 26, class_loss = 4.450562, iou_loss = 72.008652, 
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 150 Avg (IOU: 0.836025), count: 51, class_loss = 3.596705, iou_loss = 27.307808, 
v3 (iou loss, Normalizer: (iou: 0.07, obj: 1.00, cls: 1.00) Region 161 Avg (IOU: 0.859247), count: 30, class_loss = 1.494892, iou_loss = 4.325980, t
 total_bbox = 940568, rewritten_bbox = 0.075486 %
```

Figure 3. Data training process

An overview of Yolov4 architecture is shown in the Figure 4.

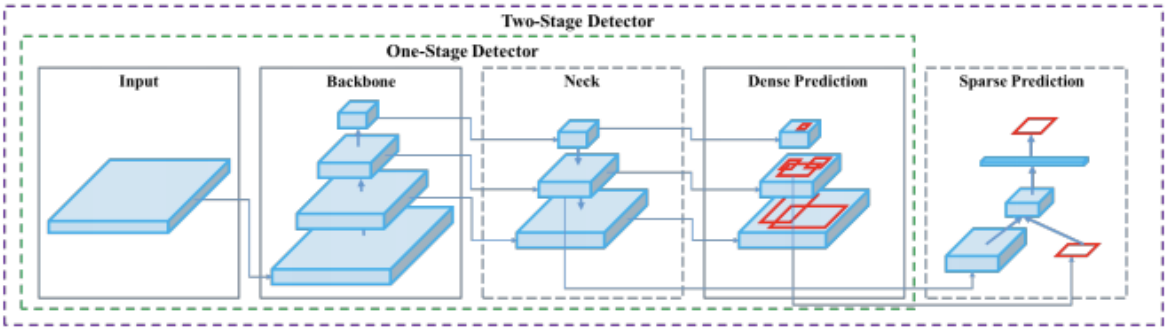


Figure 4. YOLOv4 architecture

3. RESULTS AND DISCUSSIONS

3.1. Formation of Detection Model Results

After training, the system generates a weight-based model for disease detection in cocoa beans with the YOLOv4 architecture. The model formation process starts with dataset annotation. In this study, 212 images were annotated using the Python-based LabelImg application.

Each annotated image produces one annotation file in .txt format. The file contains information about the location and class of objects in each cocoa bean image. This information is used as a reference for the model in learning the visual patterns and positions of objects to be detected. An example of the annotation process can be seen in Figure 4.

```
1 0.516667 0.589543 0.450000 0.542067
2 0.331731 0.463822 0.394872 0.323317
2 0.049840 0.482332 0.099038 0.290625
2 0.668750 0.191707 0.125321 0.330048
2 0.824679 0.559495 0.064103 0.141106
0 0.692308 0.699279 0.154487 0.145192
2 0.752564 0.113101 0.042308 0.124760
```

Figure 4. Results of one of the labeling processes

Datasets that have been labeled will be carried out the training process. In this study, the training process used a large version of the YOLOv4 algorithm or weight. Every 1000 training results from this process will be stored on Google Drive with the aim of making the training process easier. The results of the training process will form a weight-based file that is used to detect mature coconut and young coconut objects. The training process was carried out for 20 hours with iterations of 2100 epochs. The results of the training process are shown in Figure 5.

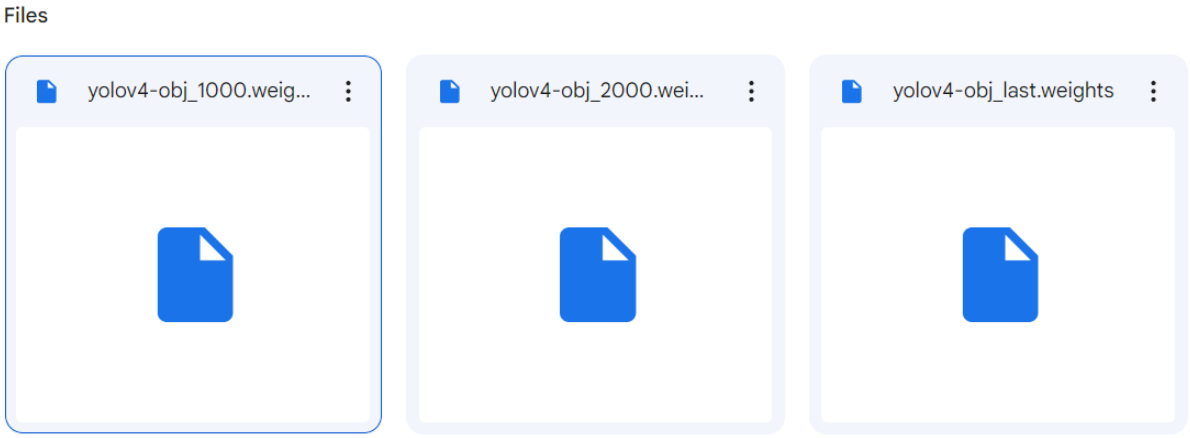


Figure 5. Training process results

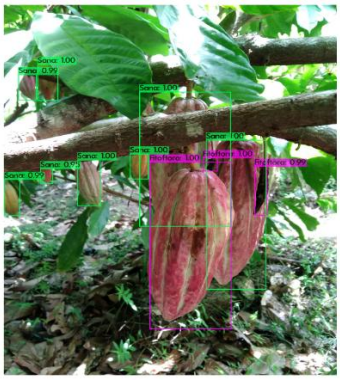
3.2. Test Results

The testing was conducted using the weight file from the previous training stage using the YOLOv4 algorithm. The testing process was carried out with the parameters batch = 1 and subdivisions = 1. The purpose of this test was to determine the model's ability to detect diseases in cocoa fruits based on the images provided.

The results of testing on several test images show that the YOLOv4 system is capable of detecting cocoa fruit objects with different classes, namely the phyto, monilia, and sana classes. Each detection result is displayed in the form of a bounding box that shows the location of the object and the class of disease detected in the image. Examples of test results can be seen in Table 2.

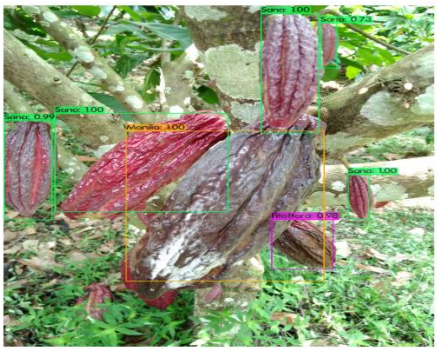
Table 2. Test results

Image Name	Data Test	Detection Results

Cocoa fruit 1		
Cocoa fruit 2		
Cocoa fruit 3		
Cocoa fruit 4		

Cocoa fruit 5

Cocoa fruit 6



Cocoa fruit 7

Cocoa fruit 8

In addition, testing was also conducted on the model's detection time. Based on the test results, the time required to detect one image was approximately 690.18 milliseconds. This detection time shows that the model is capable of performing the detection process in a relatively short time.

The test results also show that the developed model is capable of achieving a detection accuracy rate of over 97%. This accuracy value indicates that the YOLOv4 model used in this study is capable of recognizing and classifying diseases in cocoa beans well. The visualization of the detection results and detection time is shown in Figure 6.

```
Loading weights from /mydrive/yolov4/backup/yolov4-obj_2000.weights...
 seen 64, trained: 128 K-images (2 Kilo-batches_64)
Done! Loaded 162 layers from weights-file
 Detection layer: 139 - type = 28
 Detection layer: 150 - type = 28
 Detection layer: 161 - type = 28
/mydrive/images/Sana3.jpg: Predicted in 690.186000 milli-seconds.
Sana: 99%
Sana: 100%
Sana: 100%
Sana: 98%
Unable to init server: Could not connect: Connection refused
```

Figure 6. Description of results and detection time

Overall, the test results show that the trained YOLOv4 model can be used to detect diseases in cocoa beans quite well based on the images tested.

4. CONCLUSION

After testing and analyzing the system developed in this research, the following conclusions can be drawn: From the testing, it was found that the average reading distance of the RFID reader used is 5.7 cm.The performance of the coin acceptor used in this project is satisfactory. This can be observed from the test results where only valid coins, matching the reference coins placed in the coin acceptor, were accepted. The Solenoid lock requires a 12-volt power supply, whereas the digital output from the NodeMCU ESP32 is only 3 volts, thus unable to directly power the Solenoid lock. To control the Solenoid lock, an IC ULN 2803 is needed.The multiplexer circuit is employed to conserve digital pin usage on the NodeMCU ESP32. The IC used is the IC74LS151, which is an 8-to-1 multiplexer IC. The shift register circuit used in this project is aimed at reducing the digital pin usage of the NodeMCU ESP32.

REFERENCES

[1] Food and Agriculture Organization, “FAOSTAT Statistical Database – Cocoa Production.” 2022.
[2] International Cocoa Organization, “Quarterly Bulletin of Cocoa Statistics.” 2021.
[3] A. W. Susilo and others, “Management of Cocoa Pod Borer (Conopomorpha cramerella) in Indonesia,” *IOP Conf. Ser. Earth Environ. Sci.*, vol. 418, 2020.
[4] D. P. Sari, H. A. Nugroho, and N. A. Setiawan, “Identification of cocoa plant diseases based on image processing,” *J. Phys. Conf. Ser.*, vol. 1845, 2021.
[5] S. P. Mohanty, D. P. Hughes, and M. Salathe, “Using deep learning for image-based plant disease detection,” *Front. Plant Sci.*, vol. 10, 2019.
[6] Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[7] R. Szeliski, *Computer Vision: Algorithms and Applications*, 2nd ed. Springer, 2021.
[8] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” *arXiv Prepr. arXiv1804.02767*, 2018.
[9] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” *arXiv Prepr. arXiv2004.10934*, 2020.
[10] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, “Scaled-YOLOv4: Scaling Cross Stage Partial Network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.
[11] B. Min, T. Kim, and D. Shin, “applied sciences Data Augmentation Method for Plant Leaf Disease Recognition,” pp. 1–17, 2023.
[12] O. S. X. Corp, “The Effect of Improving Annotation Quality on Object Detection Datasets : A Preliminary Study,” pp. 4850–4859, 2021.