Automatic Detection of Oil Palm Growth Rate Status with YOLOv5

Desta Sandya Prasvita, Dina Chahyati, Aniati Murni Arymurthy
Faculty of Computer Science, University of Indonesia, Depok, Indonesia

***Abstract*—Oil palm plantations are essential for Indonesia as a source of foreign exchange and a provider of employment opportunities. However, large-scale land clearing is considered a cause of deforestation, which harms the environment and society. So, it is necessary to manage plantations that are sustainable and still maintain the preservation of forests and biodiversity. One solution is to apply remote sensing technology. The research was conducted to develop a multi-class detection method for the growth rate of oil palm trees, with five categories: healthy palm, dead palm, yellowish palm, mismanaged palm, and smallish palm. The deep learning-based object detection method, YOLO Version 5 (YOLOv5), is used. This study compares the YOLOv5 network models, namely YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. Parameter setting is also carried out in the BCE (Binary Cross Entropy) with Logits Loss Function to handle the problem of unbalanced data distribution in each class. The YOLOv5 model with the highest mAP value is the YOLOv5l and YOLOv5x, the YOLOv5x requires longer training time. In this study, hyperparameter optimization was also carried out using hyperparameter evolution techniques. However, it has yet to provide increased results because the experiments conducted in this study are still limited.**

***Keywords*—Automatic detection; deep learning; oil palm; YOLOv5**

I. Introduction

Palm oil consumption has witnessed a growth of approximately 9% annually in the last decade, and it is anticipated to rise further. This optimistic outlook for the palm oil industry is attributable to its increasing demand in domestic and global markets. Since 2004, the use of palm oil has occupied the highest position, with an average growth of 8% per year. In Indonesia, oil palm serves as a plantation commodity that has an important role in the economy as a source of foreign exchange and a provider of employment. Indonesia is also the world's largest palm oil producer [1]. Oil palm plantations, despite their positive impact on the economy, are often associated with negative impacts that can harm society, such as environmental degradation and conflicts with local communities. [2]. Oil palm plantations are considered a cause of deforestation that can damage biodiversity, with 2% of Indonesia's forests being turned into plantation areas [3]. Based on the background and problems related to oil palm plantations, especially in Indonesia, it is necessary to manage oil palm plantations sustainably so as not to harm the community. That is with an agricultural system oriented towards economic, social, and ecological balance. With proper agricultural practice and technology, it is believed that oil palm plantations will continue growing, with environmental sustainability maintained. Precision agriculture aims to optimize the use of resources to achieve the best results while protecting the environment with land management systems [4].

Remote sensing technology has been widely used in mining, agriculture, and plantations. This technology can be used for remote monitoring of plantation areas. Monitoring plantations manually by human labor is difficult, especially in oil palm plantations in Indonesia characterized by large plantation areas and difficult access. Methods for obtaining optimal results for detecting individual trees have also been developed. Based on previous research, three groups of approaches are used for tree detection. The three methods are classical digital image processing [5] [6] [7] [8], classical machine learning [9] [10], and deep learning. The tree-detection-based deep learning approach is divided into an approach based on CNN classification [11] [12] [13] [14] [15] [16] and another approach based on object detection [17] [18] [19] [20]. The first approach is based on classical digital image processing. There are four main stages to a classical digital image processing-based approach: image pre-processing, treetop detection, tree crown delineation, and post-processing. In the classical machine learning-based tree detection approach, feature extraction is required, which is then trained to build a classification model. The classification model is used to detect trees in images using the sliding window technique. The third approach is based on deep learning for tree detection. Deep learning methods are currently popular for object detection because of their object detection accuracy.

There are advantages and disadvantages to each technique for oil palm tree detection. The classical digital image processing approach has the advantage that it does not require a training process for the construction of classification or detection models that require high and expensive computational costs. Another advantage is the time it takes to detect quickly. However, this classic digital image processing approach has a weakness. It is not easy to use to detect, especially in the case of overlapping trees. On the other hand, the classic machine learning approach has the advantage of seeing overlapping trees. However, accuracy depends on the feature extraction used. Another weakness is that, while the detection process generally uses a sliding window technique, it takes a long time to detect trees. In addition, the sample size of the training image must have been determined when constructing the classification model. In contrast, the objects we detected were of various sizes both the classical machine learning approach and the deep learning approach using the CNN classification-based method share similar advantages and disadvantages. However, the CNN classification-based method does not necessitate feature extraction and has demonstrated superior performance. In contrast, the deep learning method based on object detection can detect objects that overlap with different trees or objects of various sizes and has a much faster detection process. This approach only requires a training process, which is costly and expensive in the computation, especially in cases with many classes.

From the background of the problem, multi-class detection with a deep learning method based on object detection in oil palm trees is state-of-the-art and still has exciting challenges. Zheng et al. [19], in their research, carried out multi-class detection of the growth rate of oil palm trees, with five categories: healthy palm, dead palm, yellowish palm, mismanaged palm, and smallish palm. However, it still has sub-optimal performance and needs improvement. One is the still low F1-score for the dead palm and mismanaged palm classes, namely 43.24% and 44.59%, respectively. The reason for this is that the class contains a relatively small sample of data compared to the quantity of data available in other categories. This study developed a research method based on the work of Zheng et al. to improve the detection results on the growth status of oil palm trees. Zheng et al. have published training datasets, which were used and compared in this research. This research has developed a model that utilizes YOLOv5 for detecting the growth status of oil palm trees, a technique that has already demonstrated success in detecting date palms [17] and tree damage [20]. This study has made several notable contributions, including: 1) utilizing the YOLOv5 method to detect tree growth status, 2) addressing the issue of unbalanced class distribution by adjusting the parameter of BCE with Logits Loss, and 3) conducting hyperparameter optimization experiments for YOLOv5.

II. Literature Review

In the development of deep learning methods based on object detection, such as R-CNN [21], R-CNN [22], faster R-CNN [23], and YOLO [24], the detection process increases in terms of both speed and accuracy. Therefore, an object detection-based deep learning approach is the state-of-the-art approach of this research. There were three primary references used in this study. The first was a research study conducted by Zheng et al. In this research, we developed a model to detect the growth rate of oil palm trees, categorizing them into healthy palms, mismanaged palms, smallish palms, yellowish palms, and dead palms. The data used was derived from unmanned aerial vehicle (UAV) images of oil palm plantation areas in South Kalimantan and Papua, Indonesia. The features in the UAV image data were red, green, and blue (RGB) features. There were five classes: dead palms, healthy palms, mismanaged palms, smallish palms, and yellowish palms. The method for detecting oil palm trees is called Multi-Class Oil Palm Detection (MOPAD). There are three main modules in MOPAD. The first module is the Refined Pyramid Feature (RPF) Module for feature extraction, including four steps: rescaling, integration, refinement, and disintegration. The second is the Multi-Level Region Proposal Network (RPN) with faster R-CNN to generate oil palm candidates. The third is the Hybrid Class-Balanced Loss Module to improve the detection of multi-class palm oil using Class-Balanced Cross-Entropy Loss (CBCEL) and Class-Balanced Smooth L1 Loss (CBSLL). An evaluation was carried out using the evaluation metrics recall, precision, and F1-score. Regarding the achieved performance, this study has produced in detecting oil palm trees at two sites, with F1-scores of 87.91% and 99.04%, precision values of 92.42% and 98.90%, and recall values of 83.82% and 99.19% [19]. However, the detection of multi-class oil palm trees was still low, with an average F1-score of 72.83% and with the most frequently detected classes being dead palms and mismanaged palms. A few challenges persisted in this study, one of which was the low F1-scores in detecting the growth of oil palm trees.

The latest object detection method widely used is YOLO. This method outperforms faster R-CNN in terms of detection speed. YOLOv5 was successfully implemented to detect tree damage due to snowfall automatically. The study utilized image data captured using drone technology from southeastern Norway, featuring RGB channels. The built model was then validated using precision (P), recall (R), $mAP@0.5$, and $mAP@[0.5,0.95]$ accuracy metrics. The validation results for P, R, $mAP@0.5$, and $mAP@[0.5,0.95]$ for all classes were 0.62, 0.61, 0.65, and 0.37. In this research, we employed the YOLOv5 method to detect tree damages caused by snow, which has not been attempted in previous studies. Furthermore, our findings demonstrate that YOLOv5 is capable of addressing the challenge of class imbalance, particularly when the percentage of damaged and dead trees is considerably lower than that of healthy trees. [20].

The YOLOv5 method was also applied to automatically detect date palms using drone imagery [17]. The data for the study were obtained using drones on agricultural land in the United Arab Emirates. The research experiment was carried out by comparing several versions of the YOLOv5 network, including YOLOv5s (small), YOLOv5m (medium), YOLOv5l (large), and YOLOv5x (extra-large). The initial model weights were the pre-training weights from training on the COCO dataset to recognize 80 classes, which were then adjusted for hyperparameters to obtain optimal parameters. The hyperparameter tuning method used was to use a genetic learning algorithm. The test results showed that YOLOv5 had a higher mAP value than SSD300, YOLOv3, and YOLOv4. The detection speeds of YOLOv5s, YOLOv5m, and YOLOv5l were also higher than those of SSD300, YOLOv3, and YOLOv4, but YOLOv5x had the longest average detection time. The YOLOv5 method, recognized for its speed, is an efficient object detection approach that surpasses other CNN-based methods such as R-CNN, fast R-CNN, and faster R-CNN in terms of speed.

This study developed a research method based on the work of Zheng et al. to improve the detection results on the growth status of oil palm trees. Zheng et al. have published training datasets, which were used and compared in this research. This research built a model for detecting the growth status of oil palm trees using YOLOv5, which has been successfully applied in detecting date palms and tree damage.

III. Method

This research went through three main stages: 1) data preparation, 2) development of the YOLOv5 model, and 3) model testing. The data used were the image data provided in a study by Zheng et al. During the data preparation stage, we regenerated the data and separated it into three parts: training data, validation data, and testing data. In the training phase, we concurrently utilized the training and validation data. We employed the training data to construct the model, while the validation data was used to test the model at each iteration and determine the best possible model. In the testing phase, we tested the model created during the training stage using the testing data and performed an evaluation. Fig. 1 illustrates the stages of the research methods.

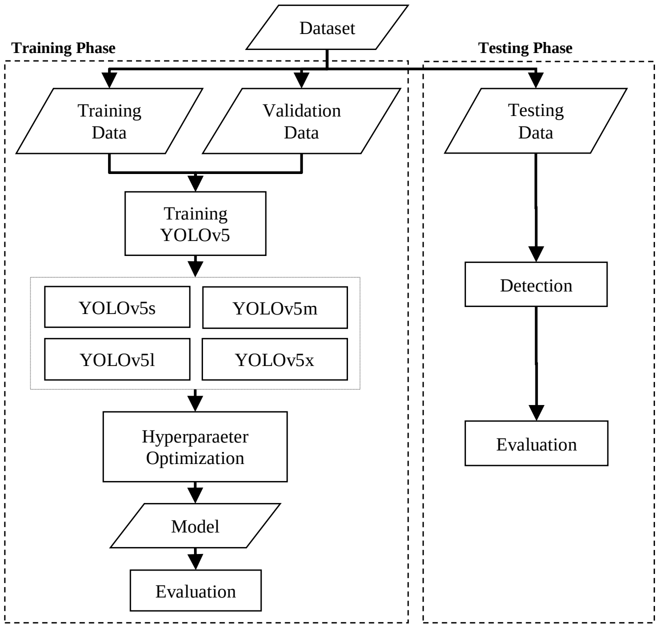

Fig. 1. Research method.

A. Data and Preparation

The dataset used in this study was composed of data from previous studies [19], available at https://github.com/rs-dl/MOPAD. The data acquisition location was Papua, Indonesia (140◦29′ 17′′E, 6◦57′ 42′′S), and the data acquisition was executed using a Skywalker X8 air bridge and a Sony a6000 camera. A spatial resolution of 8 cm with three RGB bands was used. This area had various types of land cover, such as oil palm plantations, rivers, buildings, other vegetation, etc.

For further processing for the development of the YOLOv5 model, the data obtained is carried out at the data preparation stage, namely data regeneration and reformatting. There were 2,303 image data, each measuring 1,024 x 1,024 pixels. Data regeneration was performed using the roboflow online tool. There were three stages in data regeneration: train/split test, pre-processing the data by scaling, and generating YOLOv5 format compliant data. In the train/test split process, the data were divided into training, validation, and testing data, each totaling 1,600 data, 135 data, and 136 data (70%, 15%, and 15%). The rescaling process resized the data from 1,024 x 1,024 pixels to 416 x 416 pixels. Next, necessary formatting adjustments were made to YOLOv5.

TABLE I. THE NUMBER OF IMAGES FOR TRAINING, VALIDATION, AND TESTING DATA

Type	Training Data	Validation Data	Testing Data	Total
Healthy palm	79,988	17,453	17,401	114,842
Dead palm	193	45	57	295
Mismanaged palm	407	68	51	526
Smallish palm	24,486	4,832	4,721	34,039
Yellowish palm	1,339	270	275	1,884
Total	106,413	22,668	22,505	151,586

The dataset included annotations of oil palm trees classified into five labels based on their growth rate. Specifically, the study detected five growth stages of oil palm trees, including dead palms, healthy palms, mismanaged palms, smallish palms, and yellowish palms. Overall, the dataset contained 151,586 oil palm tree annotations for all classes. Table I shows the number of annotations for each category and training, validation, and testing data.

B. The YOLOv5 Network Architecture

The object detection neural network architecture consists of three main components: the backbone extracts image features, the neck incorporates the features extracted from the previous layers, and the head predicts object classes and bounding boxes. The YOLOv5 architecture can be seen in Fig. 2.

YOLOv5 has many key components in each part of its networks, such as focus, Conv (convolution), C3, and Spatial Pyramid Pooling (SPP). The focus module divides the input image into four parallel slices (S) to create a feature map using the convolution module. The convolution module is a basic module that uses convolution operations combined with batch normalization and a leaky-ReLU activation function for feature extraction. The C3 module is designed based on a Cross-Stage Partial (CSP) connection network that is used to improve model learning capabilities. The backbone model uses the SPP module for mixing and unifying spatial features. It down-samples the input features through three parallel max pooling layers and then aggregates them to the featured initial.

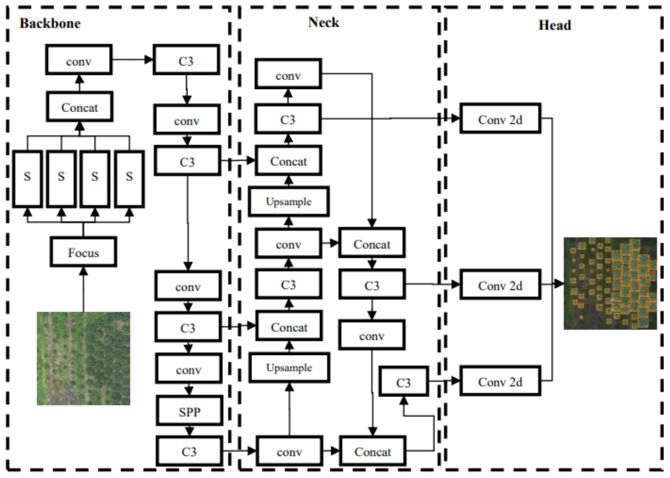

Fig. 2. The network architecture of YOLOv5.

C. Dealing with Imbalanced Data

An imbalance in class distribution can lead to the model better at predicting classes with more data and worse at predicting classes with fewer data. The dataset used in this study has a highly unbalanced distribution. Fig. 3 shows that the healthy oil palm class had a wide data distribution, as marked with a yellow bar. However, there was very little data on yellowish, mismanaged, and dead palm classes depicted in red bars.

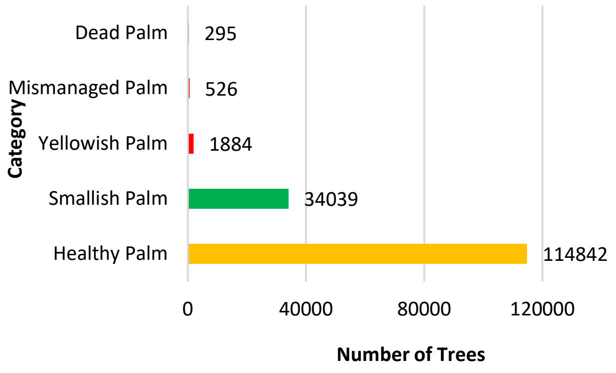

Fig. 3. Data distribution for each class.

The loss function for objectivity in YOLOv5, class probability, was calculated using BCE with Logits Loss Function. The parameters in BCE with Logits Loss were adjusted to overcome the problem of the unbalanced proportion of data in the class. Setting BCE with the Logits Loss parameter made trading of recall and precision possible by adding weight to positive examples. In the case of multi-label classification, the loss ($l_c(x,y)$) can be explained in equation (1).

$$l_c(x,y) = l_c = \{l_{1,c}, \dots, l_{N,c}\}^T,$$
$$l_{n,c} = -w_{n,c}p_c y_{n,c}.\, log\sigma(x_{n,c}) + (1 - y_{n,c}).\log 1 - \sigma xn, c \quad (1)$$

Where c is the class number, n is the sample size in the batch, and p_c is the weight of the positive for class c. For example, if the data set contains 5 positive and 200 negative examples of a class, then the pos_weight for that class must equal $\frac{200}{5} = 40$. The loss will act as if the data set contains 40×5=200 positive examples [25].

D. Training the Network

Model training was performed utilizing transfer learning from pre-trained weights of 100 epochs trained on the large Common Objects in Context (COCO) dataset. The pre-trained model was trained to recognize 80 types of objects. The training was carried out for 300 epochs to train the model to recognize objects corresponding to the dataset's five oil palm growth status categories. At each epoch, validation data were executed using the $mAP@[0.5,0.95]$ evaluation metric value. The best model was chosen with the highest $mAP@[0.5,0.95]$ value. The training image data input size was 416 x 416 pixels, with a batch size of 32.

This research built training models for four different versions of YOLOv5, namely, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. The difference between the four YOLOv5 versions lies in the number of feature extraction modules in the convolution layer. YOLOv5s has the least feature extraction modules and kernels, and YOLOv5x has the highest feature extraction and kernels. The bigger the model, the better the result, but it has more parameters, requires more CUDA memory, and takes longer to train.

YOLOv5 has around 30 hyperparameters which are used for various training settings. In addition to using hyperparameters in the pre-trained model COCO dataset, this research also optimized the hyperparameters. A hyperparameter tuning process was carried out using hyperparameter evolution [26]. Hyperparameter evolution is a powerful optimization method that uses a genetic algorithm (GA) to optimize hyperparameters in various stages. The first stage is hyperparameter initialization, which utilizes the default parameters of YOLOv5 COCO. The second stage is defining fitness, where the objective function is set to measure how well the model performs. The third stage is evolution, where the GA is utilized to find the optimal set of hyperparameters. Finally, the visualization stage provides a graphical representation of the optimization process. The metric evaluation value of $mAP@[0.5,0.95]$ determines the fitness value. The number of evolutions performed is 300, with each epoch being 10. Visualizations are performed to evaluate the evolution results. Considering the cost of the YOLO training process, the model used for hyperparameter evolution in this study was YOLOv5s only.

E. Evaluation

The model was tested using both validation and testing data. Validation data was used during each epoch of the training process to test the model, while testing data was used to evaluate the final YOLOv5 model. Testing of the object detection model was carried out using the mean average precision (mAP) evaluation metric. The mAP is the average precision (AP) in all detected classes. The AP value is only for each category. The average of 11 interpolation points on the precision-recall curve was calculated to obtain the AP value. According to the 2017 COCO challenge evaluation guidelines, mAP was calculated by the average AP over 80 object classes and all 10 IoU thresholds from 0.5 to 0.95 with a step size of 0.05, i.e., $mAP@[0.5,0.95]$. The calculation of the mAP value can be seen in equations (2) and (3).

$$mAP = \frac{1}{n}\sum_{k=1}^{n} AP_k\,,\ where\ \ n = the\ number\ of\ classes \quad (2)$$

$$AP_k = the\ AP\ of\ class\ k \quad (3)$$

To obtain the AP value, precision and recall values were needed to build a precision-recall curve. Precision is the ratio of the correct prediction for positive data compared to the overall positive predicted result. The precision value can be calculated using the following equation, TP (true positive) is the positive data correctly predicted to the positive class, and FP (false positive) is predicted as an object but false. The precision value can be obtained by using equation (4).

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

A recall is the ratio of true positive predictions to the total number of correct positive data. The recall value can be calculated using the following equation, TP (true positive) is the positive data correctly predicted to the positive class, and FN (false negative) incorrectly predicts the object there. The recall value can be obtained using equation (5).

$$Recall = \frac{TP}{TP+FN} \tag{5}$$

In object detection, TP and FP values are determined using the IoU value. The IoU value is used as the threshold. If the IoU threshold value is 0.5, and the IoU value for a prediction is 0.7, then the prediction result is expressed as TP. Meanwhile, if the IoU threshold is 0.3, it is expressed as FP. Equation (6) is used to calculate the IoU value. An illustration of the calculation of the IoU value can be seen in Fig. 4.

$$IoU = \frac{Area\ of\ Overlap}{Area\ of\ Union} \tag{6}$$

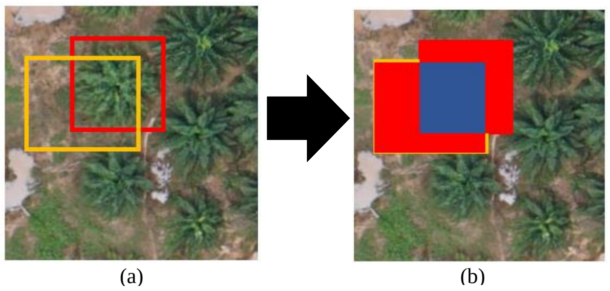

Fig. 4. Illustration of calculation of IoU value; (a) The ground truth; (b) The red color is the area of overlap, and the blue area is the area of union.

The expected model is a model with high precision and recall values. Therefore, the ideal to measure the model is F1-Score. Mathematically it can be expressed in equation (7).

$$F_1 = 2 * \frac{Precision*Recall}{Precision+Recall} \tag{7}$$

IV. Results and Discussion

A. BCE with Logits Loss Setting Parameters

One positive class weight vector was chosen as the parameter to manage imbalanced classes in the BCE with Logits Loss function. Positive classes were the class categories in the dataset that you want to detect, namely, the healthy, smallish, yellowish, mismanaged, and dead palm classes. The formula for determining class weight vectors can be seen in equation (8), where c is the detected object category. The number of positive classes is the amount of data in class c, and the number of negative classes is the amount of data in classes other than class c.

$$pos\_weight[c] = \frac{number\ of\ negative\ classes}{number\ of\ positive\ classes} \tag{8}$$

Table II provides the result of setting the BCE with Logits Loss parameter and its calculations.

TABLE II. BCE with Logits Loss Parameter Setting

Class	Count	BCEWithLogitsLoss Setting
Healthy Palm	114842	$\frac{34039+1884+526+295}{114842} = 0.32$
Smallish Palm	34039	$\frac{114842+1884+526+295}{34039} = 3.45$
Yellowish Palm	1884	$\frac{114842+34039+526+295}{1884} = 79.46$
Mismanaged Palm	526	$\frac{114842+34039+1884+295}{526} = 287.19$
Dead Palm	295	$\frac{114842+34039+1884+526}{295} = 512.85$

To set the BCE with the Logits Loss parameter in YOLOv5, it can be found in the file yolov5/utils/loss.py. There were two variables, namely BCEcls, and BCEobj. BCEobj was for objects or backgrounds, and BCEcls was for object classes. The parameter was defined in h['cls_pw'] by changing the value of the vector.

```
# Define criteria
BCEcls = nn.BCEWithLogitsLoss(pos_weight=torch.tensor([h['cls_pw']],
device=device))
BCEobj = nn.BCEWithLogitsLoss(pos_weight=torch.tensor([h['obj_pw']],
device=device))
```

The parameter was defined in h['cls_pw'] by changing the value of the vector as follows.

```
# Define criteria
BCEcls =
nn.BCEWithLogitsLoss(pos_weight=torch.tensor(torch.tensor([[512.,
287., 0.3, 3.5, 79]]), device=device))
BCEobj = nn.BCEWithLogitsLoss(pos_weight=torch.tensor([h['obj_pw']],
device=device))
```

B. Results of the Training Process

The training was conducted with two experiments to obtain the optimal model for detecting oil palm trees. The initial experiment employed a pre-trained model, whereas the subsequent one involved leveraging the evolution method to optimize hyperparameters. Notably, the training procedure was executed utilizing Google Colab Pro+.

- Training with Pre-Trained Default Models

The experiment in the training process was to compare all YOLOv5 networks, including YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. The training used 300 epochs with a batch size of 16. Table III compares evaluation metrics and training time for all YOLOv5 models.

YOLOv5l produced the highest mAP value among all YOLOv5 models, with $mAP@[0.5,0.95]$ of 0.90. Although the YOLOv5x model had a deeper network, it did not improve the results and even had the lowest average F1-score of 0.95. YOLOv5s, YOLOv5m, and YOLOv5l had a training time of 3 hours, and YOLOv5x had a training time of five hours with 300 epochs.

TABLE III. COMPARISON OF EVALUATION METRICS AND TRAINING TIME OF THE YOLOV5 MODEL

Model YOLOv5	P	R	mAP @$[0.5, 0.95]$	Average F1-score	Time
YOLOv5s	0.95	0.96	0.86	0.96	3.005 hours
YOLOv5m	0.95	0.97	0.89	0.96	3.185 hours
YOLOv5l	0.96	0.98	0.90	0.97	3.652 hours
YOLOv5x	0.95	0.95	0.90	0.95	5.550 hours

- Training with the Model by Tuning the Hyperparameter Evolution

Hyperparameter optimization experiments were carried out using the YOLOv5 model with the smallest network, namely, YOLOv5s. The number of evolutions used was 300, each using ten epochs. Fig. 5 is a visualization of the hyperparameter tuning process evolve. The yellow color indicates higher concentrations. The vertical distribution suggests that the parameter had been deactivated and not mutated.

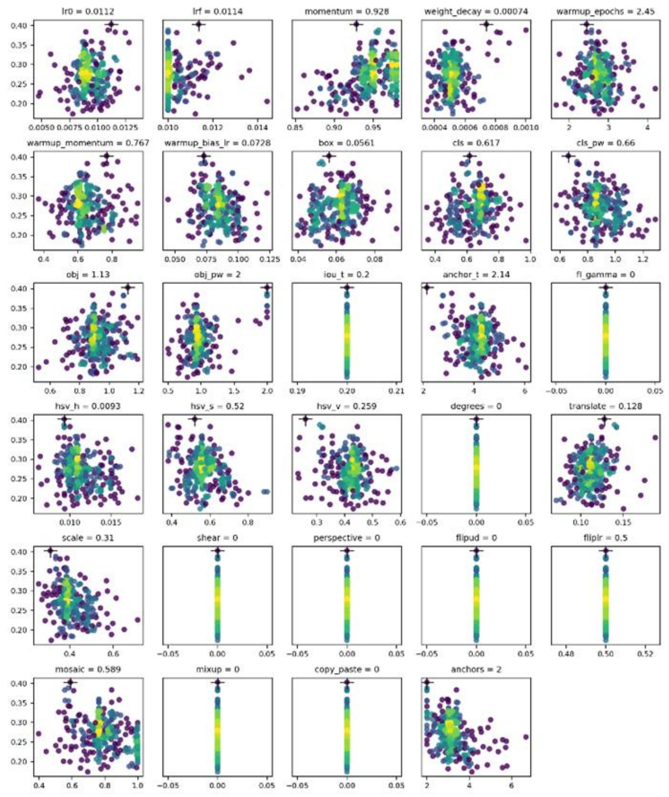

Fig. 5. Visualization of hyperparameter optimization results.

The evolved model did not improve yields compared to the default models. Fig. 6 compares the $mAP@[0.5,0.95]$ and F1-scores for detecting oil palm trees. The process of hyperparameters optimization with evolution required high costs, and evolutions required GPU processing that took days. In other words, some experimental limitations still existed at this stage, with a limited number of evolutions, namely, 300, and 10 epochs for each evolution.

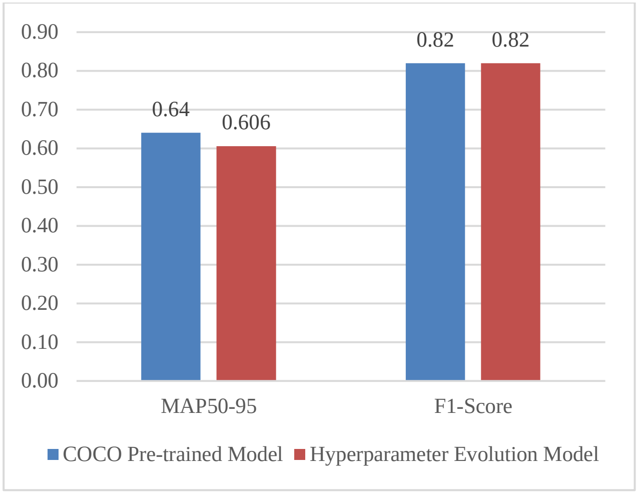

Fig. 6. Model comparison with hyperparameter evolution model and pre-trained models.

C. Model Testing Process

We compared the performance of YOLOv5 on small, medium, large, and extra-large networks using testing data. To evaluate the results, we used the F1-score and $mAP@[0.5,0.95]$ metrics on the testing data. The difference in F1 score, precisely in the mismanaged palm class, was quite far, with the lowest F1-score obtained using YOLOv5s. The F1-score for the mismanaged palm class increased significantly using YOLOv5l, in which case the increase amounted to 11.3%. Results showed that the detection of the mismanaged oil palm class had the lowest F1-score. The highest F1-score was found in the dead palm class using YOLOv5l. The F1-score comparison for each YOLO-v5 model can be seen in Fig. 7.

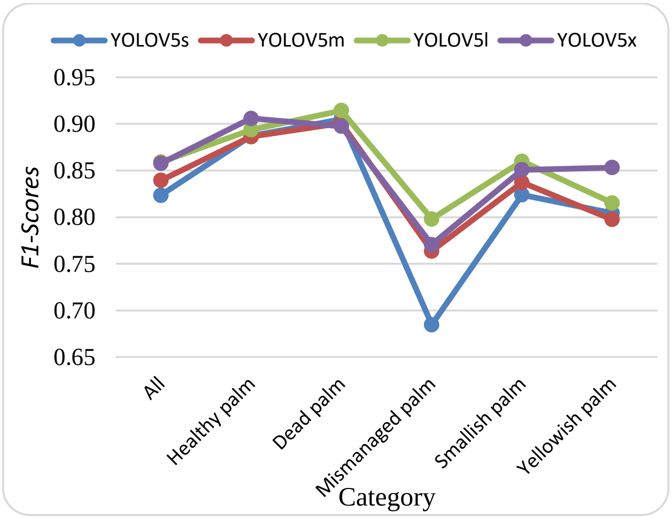

Fig. 7. Comparison of F1-Scores across YOLOv5 models.

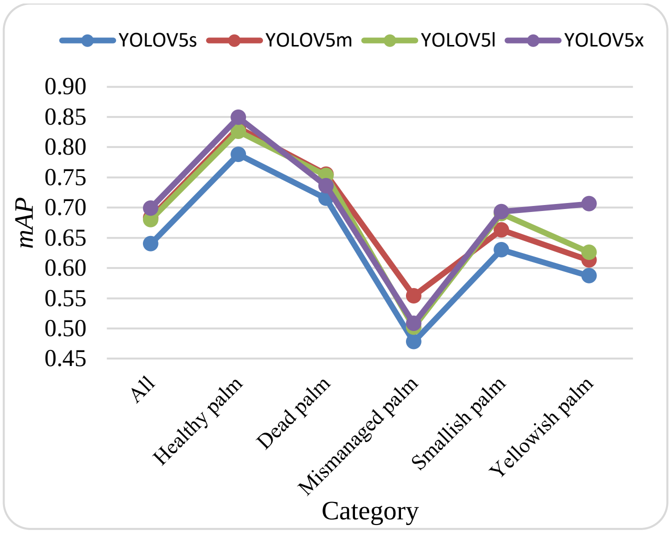

Fig. 8. Comparison of $mAP@[0.5,0.95]$ values across YOLOv5 models.

The comparison of $mAP@[0.5,0.95]$ for each YOLOv5 model can be seen in Fig. 8. The evaluation of the detection of oil palm growth rate showed that the model is suitable for detecting healthy oil palm trees. The highest $mAP@[0.5,0.95]$ value for the healthy palm oil class was 0.849, in which YOLOv5x was used. The class that was difficult to detect using YOLOv5 was the mismanaged oil palm tree class. The lowest $mAP@[0.5,0.95]$ value was 0.478, obtained with YOLOv5s, and the highest was 0.554, obtained with YOLOv5m. Detection of smallish and yellowish palms using YOLOv5 models was still difficult, but the $mAP@[0.5,0.95]$ values for these classes were better than that of the mismanaged class. For the dead palm class, the $mAP@[0.5,0.95]$ value was also quite good, which was above 0.7.

Based on the evaluation of $mAP@[0.5,0.95]$, the YOLOv5s model had the lowest scores for all classes. YOLOv5m, YOLOv5l, and YOLOv5x were in the range of $mAP@[0.5,0.95]$ evaluation values, which were not much different. YOLOv5x was sufficient to increase the value of $mAP@[0.5,0.95]$ by 0.04 compared to other YOLOv5 models. If we look at the results of the $mAP@[0.5,0.95]$ evaluation, using the YOLOv5 model with medium and large networks generated good detection results. Using the YOLOv5 extra-large network, there was not much increase in the detection results. Only the yellowish palm class saw a significant increase in the $mAP@[0.5,0.95]$ value by around 0.4.

D. Evaluation of Detection Results

The $mAP@[0.5,0.95]$ evaluation metric shows that the model was very good at detecting healthy palms and quite good at detecting dead palms. It can be seen in Fig. 9, an example of a randomly selected detected image. The detection results circled in red in Fig. 9(b), 9(c), and 9(d) are the results of mismanaged class errors (FP for mismanaged classes). The objects should have been detected as healthy oil palm trees but came out as ambiguous detection results. For instance, the objects were detected more than twice as mismanaged and healthy palms. The detection results for the mismanaged class also showed ambiguity in that one object was detected more than once, as shown in Fig. 9(a). A tree of the mismanaged palm class was detected more than once. As seen in the image with a yellow circle, the oil palm tree objects overlapped. False positive (FP) detection errors in the yellowish palm class can also be seen in Fig. 9. The yellowish class detection results are marked with a green box and a blue circle. Some objects should have been detected as members of the healthy palm class, but they were detected as part of the yellowish palm class instead. From the analysis of detection results, it was acknowledged that the model still had FP errors in detecting mismanaged and yellowish palms. From the characteristics of the image, it can be analyzed that FP errors in both the mismanaged and yellowish palm classes occurred in tree objects that overlapped each other.

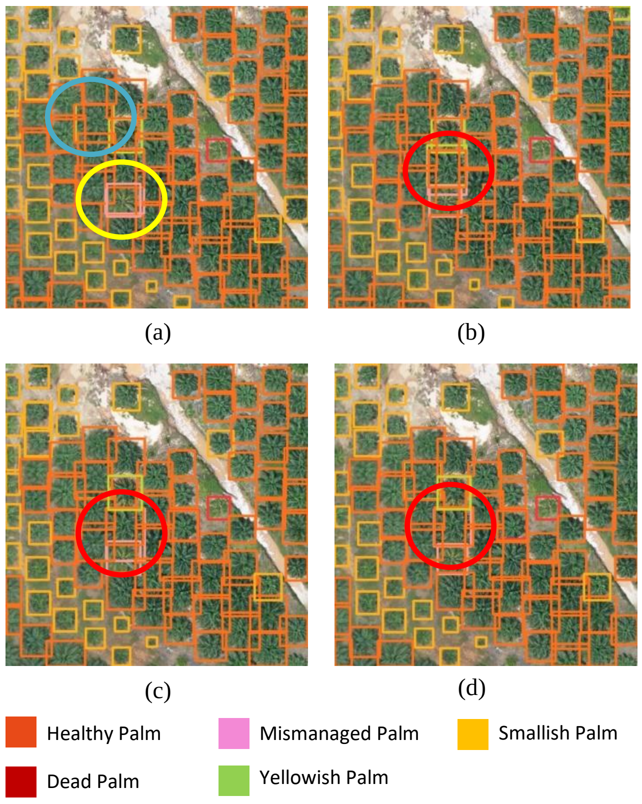

Fig. 9. Detection results for the first image sample: (a) YOLOv5s Model; (b) YOLOv5m Model; (c) YOLOv5l Model; (d) YOLOv5x Model.

Fig. 10 is an example of another detection result; there were trees in the mismanaged palm category that did not overlap, which are marked with purple bounding boxes. In contrast to the previous image, this image provides precise detection results, where oil palm trees of the mismanaged palm class were successfully detected without any false positive (FP) detection. Fig. 10 also compares YOLOv5 in each network (small, medium, large, and extra-large). In the area marked with blue circles, YOLOv5s, and YOLOv5m showed FP in the healthy palm class, where the background was detected as healthy palms. The YOLOv5l and YOLOv5x models could improve the detection results, as the red circles show.

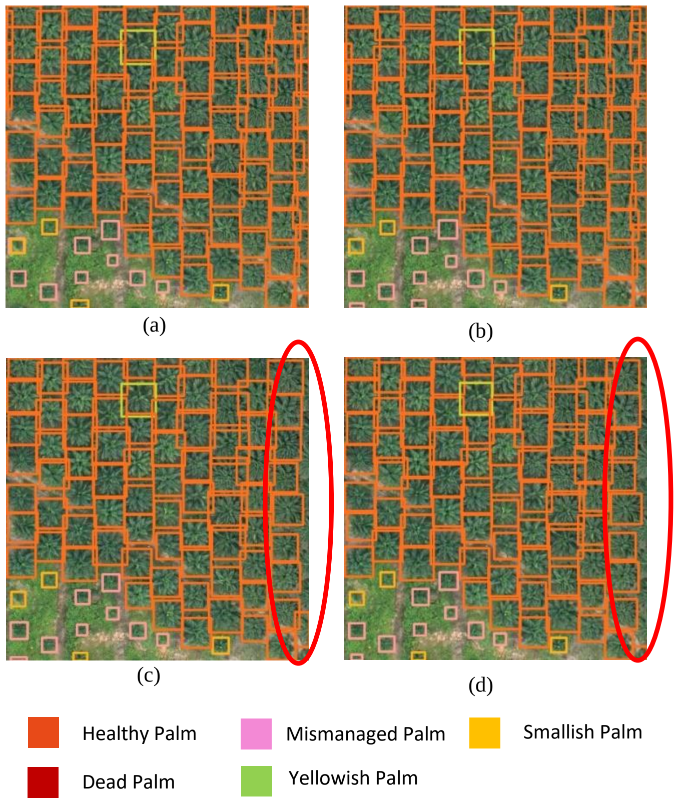

Fig. 10. Detection results for the second image sample: (a) YOLOv5s model; (b) YOLOv5m model; (c) YOLOv5l model; (d) YOLOv5x model.

E. *Comparison of the Proposed Method with Previous Research*

A comparison between the method proposed in previous studies and YOLOv5 was made. In the research conducted previously by [19], the evaluation of detection results was carried out using the F1-score. Zheng et al. named the detection method MOPAD, with an average F1-score of 72%. In addition, they also conducted experiments with other methods, such as CNN and Faster R-CNN, with an average of 28% and 56% for the two methods. As seen in Table IV, this study has improved F1-score results even with the YOLOv5s network, with an average F1-score of 72%. The highest F1-score was obtained using YOLOv5x, with an average F1-score of 86%.

The highest average F1-Score is the YOLOv5x model, but the difference in the average F1-Score is only 1% compared to the YOLOv5l. The training time for the YOLOv5x model is longer than the YOLOv5l, with a difference in training time of about two hours. It can be seen in Table IV. The analysis results show that the best oil palm growth rate detection model is YOLOv5l. Besides having a faster training time, YOLOv5l outperforms its competitors in detecting five different classes related to the growth rate of oil palm trees. Specifically, three categories - dead palm, mismanaged palm, and smallish palm - achieve the highest F1-score with YOLOv5l. The evaluation of each class was as follows: healthy palms on YOLOv5x, dead palms on YOLOv5l, mismanaged palms on YOLOv5l, smallish palms on YOLOv5l, and yellowish palms on MOPAD.

TABLE IV. COMPARISON OF THE PROPOSED METHOD WITH PREVIOUS RESEARCH

Method	Healthy	Dead	Mis-managed	Smallish	Yellowish	Average F1-score
CNN (ResNet-101) [19]	0.75	0.07	0.03	0.36	0.19	0.28
Faster R-CNN [19]	0.90	0.06	0.42	0.66	0.74	0.56
MOPAD [19]	**0.91**	0.55	0.51	0.77	**0.88**	0.72
YOLOv5s	0.89	0.90	0.68	0.82	0.80	0.82
YOLOv5m	0.89	0.90	0.76	0.84	0.79	0.84
YOLOv5l	0.89	**0.91**	**0.80**	**0.86**	0.81	0.85
YOLOv5x	**0.91**	0.90	0.77	0.85	0.85	**0.86**

V. CONCLUSION

This study found several contributions, including: 1) The application of the YOLOv5 method for detecting the growth level of oil palm trees. 2) A comparison of the YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x models. 3) Handling unbalanced class distributions by setting positive class vector parameters in the BCE with the Logits Loss function. 4) Conducting experiments to optimize hyperparameters with evolutions.

Based on the experiments conducted in this study, the YOLOv5 model improved the results of detecting the growth level of oil palm trees. The highest F1-score was obtained with the YOLOv5x model, but it took the longest learning time. The YOLOv5l model is the best because the training time is shorter, and the difference in F1 scores is slightly smaller than the YOLOv5x model by 1%. In addition to outperforming other YOLOv5 models' F1-score, the evaluation results for each class show YOLOv5l achieved the highest F1-score for 3 out of 5 classes. The detection results showed that the model still had errors in predicting trees of the mismanaged and yellowish palm classes, and the FP value for overlapping trees was still relatively high.

Setting positive class vector parameters in the BCE with the Logits Loss function could solve the unbalanced class distribution problem; where there was an imbalance in the class distribution, there the imbalance was extreme. Still, adjusting BCE with Logits Loss function parameters could produce an evaluation value for each class that was still quite good. Hyperparameter optimization by evolutions required high costs, where the training process could take days to complete. This study was limited to hyperparameter optimization experiments on the YOLOv5s model with 300 evolutions and ten epochs for each evolution. The hyperparameter optimization with evolutions results did not show a significant increase in the model.

This study applied a deep learning method based on object detection, a state-of-the-art method in oil palm tree detection research. This research still faced challenges, especially in detecting multi-class oil palm trees. It is insufficient to only feature RGB images for detecting tree growth and health level, providing only color and texture information. To better detect the growth level of oil palm trees, it is advisable to add remote sensing sensors.

ACKNOWLEDGMENT

This research is partially supported by an internal publication grant from the Faculty of Computer Science, University of Indonesia.

REFERENCES

[1] Kementrian Perindustrian RI, Tantangan dan Prospek Hilirisasi Sawit Nasional Analisis Pembangunan Industri. 2021.

[2] Ngadi and M. Noveria, “Keberlanjutan Perkebunan Kelapa Sawit di Indonesia dan Prospek Pengembangan Perbatasan,” J. Masy. Indones., vol. 43, no. 1, pp. 95–111, 2017.

[3] E. Meijaard et al., Kelapa sawit dan Keanekaragaman Hayati Analisis situasi oleh Satuan Tugas Kelapa Sawit IUCN. 2018.

[4] E. N. Ginting and D. Wiratmoko, “Potensi dan Tantangan Penerapan Precision Farming dalam Upaya Membangun Perkebunan Kelapa Sawit yang Berkelanjutan,” War. PPKS, vol. 26, no. 2, pp. 55–65, 2021.

[5] H. Z. M. Shafri, N. Hamdan, and M. I. Saripan, “Semi-automatic detection and counting of oil palm trees from high spatial resolution airborne imagery,” Int. J. Remote Sens., vol. 32, no. 8, pp. 2095–2115, 2011, doi: 10.1080/01431161003662928.

[6] J. Yang, Z. Kang, S. Cheng, Z. Yang, and P. H. Akwensi, “An Individual Tree Segmentation Method Based on Watershed Algorithm and Three-Dimensional Spatial Distribution Analysis from Airborne LiDAR Point Clouds,” IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens., vol. 13, pp. 1055–1067, 2020, doi: 10.1109/JSTARS.2020.2979369.

[7] T. Yun et al., “Individual tree crown segmentation from airborne LiDAR data using a novel Gaussian filter and energy function minimization-based approach,” Remote Sens. Environ., vol. 256, no. December 2020, p. 112307, 2021, doi: 10.1016/j.rse.2021.112307.

[8] A. Harikumar, P. D’Odorico, and I. Ensminger, “A Fuzzy Approach to Individual Tree Crown Delineation in Uav Based Photogrammetric Multispectral Data,” in International Geoscience and Remote Sensing Symposium, 2020, pp. 4132–4135.

[9] Y. Wang, X. Zhu, and B. Wu, “Automatic detection of individual oil palm trees from UAV images using HOG features and an SVM classifier,” Int. J. Remote Sens., vol. 40, no. 19, pp. 7356–7370, 2019, doi: 10.1080/01431161.2018.1513669.

[10] A. P. D. Corte et al., “Forest inventory with high-density UAV-Lidar: Machine learning approaches for predicting individual tree attributes,” Comput. Electron. Agric., vol. 179, no. April, p. 105815, 2020, doi: 10.1016/j.compag.2020.105815.

[11] D. S. Prasvita, M. M. Santoni, R. Wirawan, and N. Trihastuti, “Klasifikasi Pohon Kelapa Sawit Pada Data Fusi Citra Lidar Dan Foto Udara Menggunakan Convolutional Neural Network,” JIPI (Jurnal Ilm. Penelit. dan Pembelajaran Inform., vol. 6, no. 2, pp. 406–415, 2021, doi: 10.29100/jipi.v6i2.2437.

[12] W. Li, H. Fu, L. Yu, and A. Cracknell, “Deep learning based oil palm tree detection and counting for high-resolution remote sensing images,” Remote Sens., vol. 9, no. 1, 2017, doi: 10.3390/rs9010022.

[13] W. Li, H. Fu, and L. Yu, “Deep convolutional neural network based large-scale oil palm tree detection for high-resolution remote sensing images,” in International Geoscience and Remote Sensing Symposium (IGARSS), 2017, vol. 2017-July, pp. 846–849, doi: 10.1109/IGARSS.2017.8127085.

[14] N. A. Mubin, E. Nadarajoo, H. Z. M. Shafri, and A. Hamedianfar, “Young and mature oil palm tree detection and counting using convolutional neural network deep learning method,” Int. J. Remote Sens., vol. 40, no. 19, pp. 7500–7515, 2019, doi: 10.1080/01431161.2019.1569282.

[15] W. Li, R. Dong, H. Fu, and L. Yu, “Large-Scale Oil Palm Tree Detection from High-Resolution Satellite Images Using Two-Stage Convolutional Neural Networks,” Remote Sens., vol. 11, no. 1, 2019, doi: 10.3390/rs11010011.

[16] S. R. Aliandra and D. S. Prasvita, “Application of Median Filter Method for Classification of Oil Palm Tree on LiDAR Images,” pp. 441–444, 2022.

[17] T. Jintasuttisak, E. Edirisinghe, and A. Elbattay, “Deep neural network based date palm tree detection in drone imagery,” Comput. Electron. Agric., vol. 192, no. November 2021, p. 106560, 2022, doi: 10.1016/j.compag.2021.106560.

[18] Z. Hao et al., “Automated tree-crown and height detection in a young forest plantation using mask region-based convolutional neural network (Mask R-CNN),” ISPRS J. Photogramm. Remote Sens., vol. 178, no. May, pp. 112–123, 2021, doi: 10.1016/j.isprsjprs.2021.06.003.

[19] J. Zheng et al., “Growing status observation for oil palm trees using Unmanned Aerial Vehicle (UAV) images,” ISPRS J. Photogramm. Remote Sens., vol. 173, no. August 2020, pp. 95–121, 2021, doi: 10.1016/j.isprsjprs.2021.01.008.

[20] S. Puliti and R. Astrup, “Automatic detection of snow breakage at single tree level using YOLOv5 applied to UAV imagery,” Int. J. Appl. Earth Obs. Geoinf., vol. 112, no. August, p. 102946, 2022, doi: 10.1016/j.jag.2022.102946.

[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 580–587, 2014, doi: 10.1109/CVPR.2014.81.

[22] R. Girshick, “Fast R-CNN,” Proc. IEEE Int. Conf. Comput. Vis., vol. 2015 Inter, pp. 1440–1448, 2015, doi: 10.1109/ICCV.2015.169.

[23] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, 2017, doi: 10.1109/TPAMI.2016.2577031.

[24] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-Decem, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.

[25] The PyTorch Foundation, “BCEWITHLOGITSLOSS,” 2022. https://pytorch.org/docs/stable/generated/torch.nn.BCEWithLogitsLoss.html.

[26] G. Jocher, “Hyperparameter Evolution,” https://docs.ultralytics.com/, 2022. https://docs.ultralytics.com/tutorials/hyperparameter-evolution/.