An efficient strawberry segmentation model based on Mask R-CNN and TensorRT

1st Anthony Crespo
School of Mathematical and Computational Sciences
Yachay Tech University
San Miguel de Urcuquí, Ecuador
brian.crespo@yachaytech.edu.ec

2nd Claudia Moncada
School of Mathematical and Computational Sciences
Yachay Tech University
San Miguel de Urcuquí, Ecuador
claudia.moncada@yachaytech.edu.ec

3rd Fabricio Crespo
School of Mathematical and Computational Sciences
name of organization (of Aff.)
City, Country
email address or ORCID

4th Manuel Eugenio Morocho-Cayamcela
School of Mathematical and Computational Sciences
Yachay Tech University
San Miguel de Urcuquí, Ecuador
mmorocho@yachaytech.edu.ec

***Abstract*—Fruit instance segmentation plays a crucial role in autonomous harvesting systems. This process involves the identification and separation of individual fruits within a crop, allowing a more efficient and accurate harvesting process. Although deep learning techniques such as detection and segmentation have shown potential for this task, the complexity of these models leads to difficulty in their implementation in real-time systems. For this reason, a model capable of performing adequately in real-time, while also having good precision is of great interest. In this way, the objective of this work is to use the Mask R-CNN to perform instance segmentation in the StrawDI_Db1 database, which contains a total of 3100 images of strawberries in real crops and provides ground-truth segmentations. Unlike the approaches, we used the NVIDIA Tensor RT framework to optimize the model for real-time use. We focus on three metrics: mAP, FPS, and size in MB. As a result, we obtained a non-optimized model with a performance of 83.32 mAP and 4 FPS, with a size of 335 MB which after the TensorRT optimization process achieved values of 82.84 mAP, 21.24 FPS, with a size of 48 MB, which surpasses those obtained by the considered benchmark paper.**

***Index Terms*—Deep learning, Image segmentation, Mask R-CNN, Strawberry segmentation, Computer vision.**

I. Introduction

Fruit segmentation has become a very important activity to avoid possible diseases and improve harvesting times [1], [2]. As a consequence, the focus of food harvesting is shifting to process automation, and the strawberry industry is no exception. Particularly, one of the most prominent problems related to strawberry cultivation is the high cost of picking. This activity requires a large amount of labor, which increases costs [3], [4]. This means that a large amount of time is required to complete this task, which translates into limited daily production.

Although automating this process is a solution for this problem, there are certain factors such as lighting, overlapping fruit, and occlusion of fruit by branches or leaves, among others that influence the detection of the objects and therefore represent a considerable difficulty to complete the task of segmentation effectively. However, deep learning techniques offer a promising solution to overcome these obstacles [5]. Specifically, fruit segmentation aids in the automation of the sorting process by separating fruit into categories based on size, shape, and ripeness [6] [7]. Segmentation is a technique that helps to identify and segment each fruit in the image as a separate instance. It means that you obtain a precise segmentation of each object in the image and not only the regions that contain certain objects. This is helpful to obtain more precise and detailed data. Also is a good option to recognize the appearance and size of the fruits in order to work with variations of them. Thus, this helps with automation through the use of automated harvesting systems because you need the most detailed data possible for effective execution. That helps to increase productivity, which translates into the reduction of labor required [8] [9].

Although there are a lot of recent works that tackle this segmentation task, but the proposed implementation may be a favorable idea to be implemented in the future into embedded systems that work in real time. Also, the computing time required is another important factor that need to taken into account in order to have the best final performance [10]. For this reason, a methodology that allows obtaining adequate precision, while its implementation in real-time systems is viable, results of great interest.

In this paper, we propose an improvement in terms of the mean average precision (mAP) metric, which helps to measure the precision of the localization and size of the masks, and frames per second (fps), which is the amount of images that the model can process in a second, both obtained by the benchmark paper on the StrawDI Db1 dataset [10]. For this purpose, we will perform a Mask R-CNN architecture and finetune the model to obtain high efficiency. Additionally, the TensorRT framework by NVIDIA will be used in order to optimize the model to make it usable in real-time strawberry detection systems. Also, it is used in order not to compromise

a large amount of precision in the model.

II. Related Works

Over the years, several techniques have been applied for the segmentation of fruits, which have had encouraging results for this task. The segmentation activity in its beginnings used techniques such as the ones mentioned by Mohammed *et al.* [11], who proposed a recognition method based on color threshold and k-means clustering in order to identify ripe and unripe strawberries. Also, Durand-Petiteville *et al.* [12] proposed an image processing algorithm based on the color space R-G for image segmentation. This technique is based on the subtraction of the green layer from the red one. As well, Changqi [13] *et al.* proposed a segmentation algorithm for strawberry disease identification based on methods like gray morphology, OTSU, and mean shift segmentation. At the end, they implemented a Support Vector Machines (SVM) classifier to make the label prediction.

In recent years, deep learning-based techniques have proven to be effective in fruit segmentation and have gained ground in this field. Particularly, Mask R-CNN has gained a place as one of the most used architectures for instance segmentation. Tao *et al.* [14] proposed a tea buds and leaves segmentation method based on a Mask R-CNN in conjunction with a ResNeXt, a combination of ResNet and Inception, as a backbone in feature extraction. Additionally, Zhang *et al.* [15] and Costa *et al.* [16] made use of Mask R-CNN using a ResNet50 for the backbone for ripe tomatoes and pecan nut segmentation respectively. Moreover, Jaesu *et al.* [17] use a Mask R-CNN for the task of tomato segmentation based on their mass and dimension values using a ResNet101 for feature extraction to improve speed and accuracy. Finally, Jia *et al.* in [18] used a Mask R-CNN with a Residual Network (ResNet) combined with a Densely Connected Convolutional Network (DenseNet) as the backbone for apple recognition and segmentation.

In the case of the strawberry, there are many works on instance segmentation focused on this particular fruit. Usman *et al.* [3] proposed a Mask R-CNN based model to obtain mask predictions for strawberry disease segmentation achieving a final mAP of 82.43% on the Strawberry Disease Detection Dataset. Yuanyue *et al.* [19] presented a Mask-RCNN-based instance segmentation method for strawberries using ResNet CNN for feature extraction obtaining a final mean average precision (mAP) of 72 % and F1 score of 88%. Similarly, Yu *et al.* [20] used a Mask R-CNN, with a ResNet50 backbone for ripe strawberry segmentation, obtaining a detection precision rate of 95.78%, and a mean intersection over union (MIoU) rate of 89.85%. Borrero *et al.* [10] proposed a modified version of Mask R-CNN for strawberries instance segmentation for real-time implementation. The new architecture reduced the complexity of the original Mask-RCNN reaching an mAP of 43.85% and an average of 10 fps over the StrawDI_DB1 dataset.

Lastly, there are some recent works that made use of TensorRT for model optimization. Wang *et al.* [21] used YOLOv4 to detect dirty eggs and accelerated the detection process with no accuracy loss using TensorRT. They reached an accuracy of 75.88% with a speed up to 2.3 fps. Tao *et al.* [22] proposed an improved CenterNet model to detect the presence of drones in the video frames accelerating the inference using TensorRT. They concluded that the models accelerated with this framework achieved a higher speed and maintained their detection precision. Finally, Lan *et al.* [23] proposed MobileNetV2-UNet and FFB-BiSeNetV2 models, based on U-Net and BiSeNetV2 for rice weeds semantic segmentation. After TensorRT optimization. their models reached an inference speed 75.37% and 31.7% faster than before optimization.

III. System Model and Methodology

The dataset, architecture, evaluation metrics, and optimization method used in this study will be explained in the following subsections. The steps followed in this study can be seen in Figure 1.

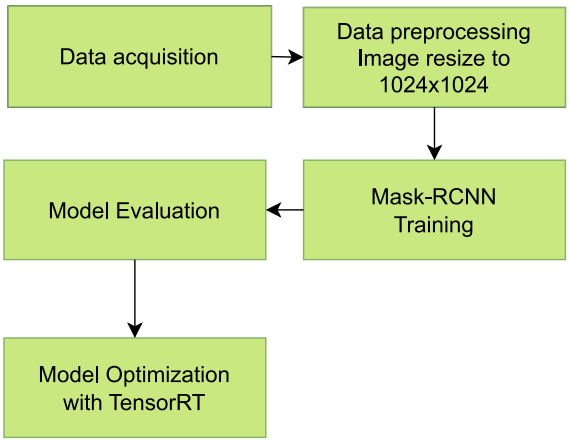

Figure 1: Flow diagram of the segmentation model based on Mask R-CNN and TensorRT

A. StrawberryDI Dataset

The Strawberry Digital Images (StrawberryDI) dataset [10] is made up of images of strawberries under different conditions collected from 20 different crops. The dataset is conformed of a total of 8000 images taken in real production conditions during the harvesting campaigns of the fruit. The images were acquired using a Samsung Galaxy S7 Edge under different lighting and distance conditions. Furthermore, each image presents differences in the size and shape of strawberries, groupings, and occlusions that are typical of the natural environment of these crops.

B. StrawDI Db1 Dataset

From the StrawberryDI dataset, a total of 3100 images were randomly selected to generate the StrawDI Db1 dataset, which will be used in this study. This dataset contains 2800, 100, and 200 images for training, validation, and testing respectively. The images are in a PNG format and their resolution is 1008

x 756 pixels. The labels are given as segmentation masks for each one of the images as seen in Figure 2.

A problem related to the use of segmentation mask labels is the high RAM consumption during training. So, to perform this demanding task, it is necessary to have a high RAM capacity becoming a limiting factor for many users. In order to solve this issue, we transformed the segmentation mask annotations into COCO format annotations. COCO is a standard and widely known format used for different computer vision (CV) tasks. Transforming the annotations to COCO format allows us to use the COCO API to make model evaluation computing the mAP metric in an easier way.

(a) Original images (b) Annotated images

Figure 2: Original and annotated images.

C. Proposed Model

1) Mask R-CNN: One of the most widely used architectures in instance segmentation is the Mask R-CNN. This is a model that is a combination of the Faster-RCNN and Fully Connected Network (FCN) to obtain an output of class and boxes. In the final model, a backbone of a ResNet50 and a Feature Pyramid Network (FPN) was used for feature extraction given an input image. The FPN helps to extract characteristics from different scales. The first step is to take the characteristics from the ResNet50 which have the bigger convolutional level. After this, the FPN uses a convolutional layer to reduce the size of those characteristics. Those layers are merged from the different levels and repeat that step consecutively to have a new layer with more robust characteristics in order to perform the detection and segmentation part. Two stages are followed for the implementation of this model. The first one is based on a Region Proposal Network (RPN) which is responsible for performing the bounding box prediction based on the anchor boxes. Also detects object regions from the multi-scale features. The second stage is based on the R-CNN detector in charge of classification and segmentation calculation at the pixel level. Based on the RPN information and have the characteristics of the image, regions of interest (RoI) are generated in the image where the model predicts that there is an object. For each RoI, the network is divided into three branches: one in charge of generating the bounding box, another to predict the class to which it belongs and with a specific probability, and finally the multi-branch network that generates a pixel-level mask, classification, and bounding box respectively. For the segmentation part, the model predicts a binary mask for each object detected during the process. The final output is a list of detections that include the class for the object, the location of the bounding box, and the binary mask for the object. A representation of the architecture of the Mask R-CNN network is given in Figure 3.

D. Loss Function

Three loss functions are defined for the Mask R-CNN: the classification loss (L_{cls}), the bounding box regression loss (L_{box}), both defined the same as in the Faster R-CNN [24] [25], and the mask loss (L_{mask}) which is only defined for the ground truth class. Adding them, then the general multitask loss function is defined as:

$$L = L_{cls} + L_{box} + L_{mask} \tag{1}$$

where,

$$L_{cls} = -log(P_T) \tag{2}$$

Where P is the predicted probability distribution of classes, and T is the index for the true class.

$$L_{box}(t_i, t_i^*) = \sum_{i \in \{x,y,w,h\}} smooth_{L_1}(t_i - t_i^*) \tag{3}$$

where $smooth_{L_1}$ is a smooth loss function, which is defined as follows:

$$smooth_{L_1}(x) = \begin{cases} 0.5x^2 & if \quad |x| < 1 \\ |x| - 0.5 & otherwise \end{cases} \tag{4}$$

We can define t_i as the vector of coordinates of the predicted bounding box, and define t_i^* as the vector of coordinates of the true bounding box. Also $\{x, y, w, h\}$ are defined as the coordinates of the bounding box center.

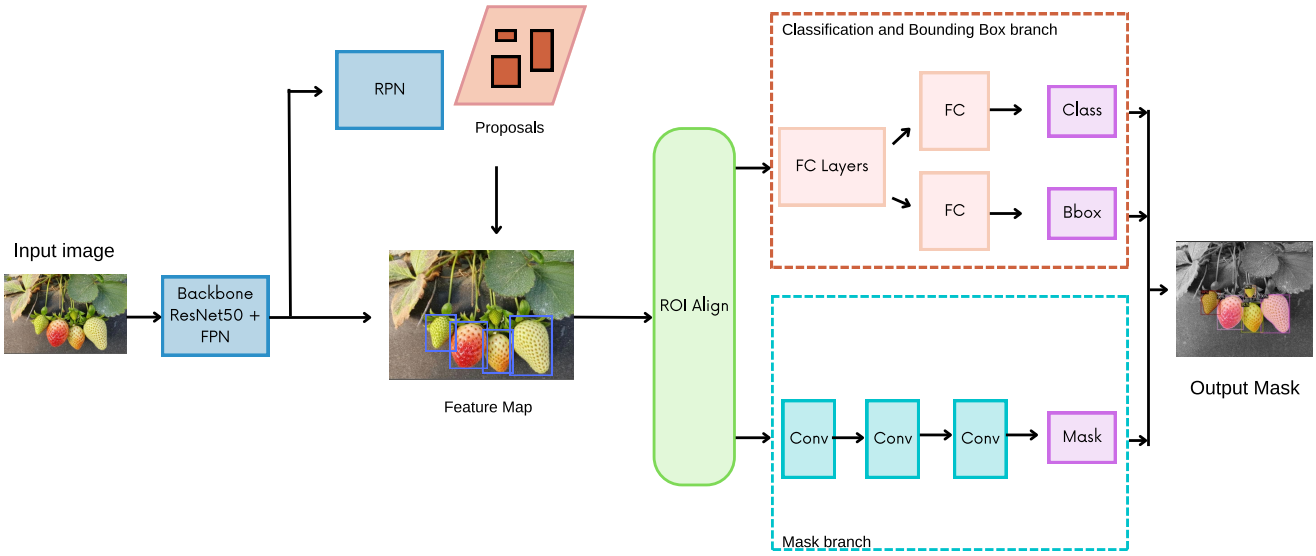

Figure 3: Mask R-CNN Architecture.

$$L_{mask} = -(m^* log(m) + (1-m) log(1-m)) \quad (5)$$

We can define m as the predicted segmentation mask and m^* as the truth segmentation mask. It is used to estimate the difference between the segmentation mask predicted by the model and the real segmentation mask defined for each region of interest (RoI) for the specific image.

E. Evaluation Metrics

The proposed model is evaluated using the mean average precision (mAP), which is given by the average precision (AP). It is commonly used in instance segmentation tasks. AP represents the area under the precision-recall curve, so its values are between 0 and 1. To determine this it is necessary to first calculate the precision (6) and recall (7). The precision represents the number of correct predictions while the recall represents the proportion of all true positives that are correct.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \quad (6)$$

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \quad (7)$$

Another intermediate step is the calculation of the Intersection over union (IoU) (8), which represents the overlap of the calculated bounding mask coordinates with the actual bounding mask. A high IoU value represents that the predicted coordinates are very similar to the actual coordinates.

$$IoU = \frac{AreaofIntersection}{AreaofUnion} \quad (8)$$

Finally, mAP (mean average precision) (9) is the average of AP. In some context, we compute the AP for each class and average them. For the calculation of the AP we use IoU thresholds ranging from 0.5 to 0.95 with step frequency of 0.05.

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \quad (9)$$

For the calculation of the AP we use a set of IoU thresholds for each class, then we compute the average of all AP values. We first computed the AP for each image and then computed the mean of all the values in order to obtain the final mAP.

F. Size Conditions

The variable sizes, respectively small, medium and large in the context of the Microsoft COCO (Common Objects in Context) dataset refer to three subsets of images containing objects of different sizes. These subsets are defined according to the area of the bounding box around the objects in the images. These subsets are useful for evaluating the performance of object detection models at different scales and levels of detail. The sizes are in accordance with the following conditions:

- Small: This subset includes images containing at least one object whose bounding box has an area of between 0 and 32^2 pixels.
- Medium: This subset includes images containing at least one object whose bounding box has an area of between 32^2 and 96^2 pixels.
- Large: This subset includes images that contain at least one object whose bounding box has an area larger than 96^2 pixels.

G. Hyperparameters

The Mask R-CNN model was trained with 4000 iterations starting from a learning rate of 0.01. The learning rate decays

every 1000 iterations with a decay factor of 0.1. The total amount of decays is 3 until a learning rate of 0.00001. In addition, the Stochastic Gradient Descent (SGD) optimizer with a momentum equal to 0.9 was used.

H. Framework and Hardware Acceleration

1) Machine learning framework: The training and tests of the models were performed using the Detectron2 AI Research library which uses Pytorch deep learning framework [26]. Detectron2 enables us to implement state-of-the-art models for detection and segmentation CV tasks.

2) Optimization Software: NVIDIA TensorRT: In order to optimize the Mask R-CNN model with the purpose to deploy it in real-life applications, it was useful to use NVIDIA TensorRT which is a Software Development Kit (SDK) that offers a deep learning inference optimizer to get high-performance preserving accuracy. It enables trained neural network models to be optimized, making it possible to deploy the model on embedded systems. The optimization is carried out using different quantization methods such as FP32, FP16, and INT8. Some TensorRT approaches are the fusion of layers and tensors, weight and activation precision calibration, dynamic tensor memory, and other ones described in the official TensorRT documentation [27]. A limitation of TensorRT is that there is no way to do the optimization directly from the Mask R-CNN produced by Detectron2 to the optimized TensorRT. Therefore, it is necessary to transform the Detectron2 model into an Open Natural Network Exchange (ONNX) model. ONNX allows the conversion of Artificial Intelligence (AI) models between different machine learning (ML) frameworks. So, when the model is in ONNX format, we can convert it to a TensorRT-optimized model.

3) Hardware Acceleration: The hardware acceleration was in charge of an MSI GeForce RTX 4090 Trio graphical processing unit (GPU) with 24 GB of RAM and 21 Gbps, with a 5.40GHz Intel ® Core i7-13700K CPU.

IV. Results and Discussion

This section provides details on the main results obtained after the experimentation phase with the proposed model. First, the implementation details and the criteria followed to improve the network during the training stage are specified. Subsequently, the evaluation results are broken down and the results of the proposed methodology are compared with those of the selected references.

A. Testing and evaluation of the Mask R-CNN from Detectron2

The obtained model is capable to detect and segment strawberry fruits, assigning them a bounding box, a label class (in this particular case, strawberry class), and the respective instance segmentation mask. Furthermore, it is able to deal with strawberries that are affected by overlapping, occlusion by leaves or branches, illumination differences, and other important determining factors for instance segmentation.

The performance of this segmentation model is evaluated using the mAP metric (imported from the COCO API) with different IoU threshold values. The model gives a prediction as correct if the calculated IoU value is greater or equal than the IoU threshold. Conversely, if the IoU value is less than the threshold, the prediction is considered incorrect. Additionally, mAP can be evaluated across three scales: small, medium, and large. All of them represent different instance sizes. It is useful when it is necessary to evaluate the performance of the model according to the size of the object.

Table I summarizes the evaluation metrics results. We observe that the mAP is 83.32 which is a greater than the 43.85 obtained by Borrero *et al.* [10]. Moreover, we see performance over 88.12 in the other mAP metrics except, for the mAP_{small}. This can be due to the complexity of making image segmentation over small strawberry fruits occluded or presented in the background of the images. Additionally, Figure 4 shows the segmentation masks applied by our model over the strawberry fruits present in the images. In the examples, we can see that the masks are applied with great precision avoiding marking pixels that do not correspond to strawberry fruit. Even when a tail crosses the fruit, the masks are only applied to the fruit and not to the tail.

Table I: Mask RCNN results on the test set for segmentation according to the mAP metric.

Metric	Value (%)
mAP	83.32%
mAP_{50}	96.60%
mAP_{75}	90.50%
mAP_{small}	50.93%
mAP_{medium}	88.12%
mAP_{large}	96.86%

Figure 4: Examples of segmentations performed by the Mask R-CNN model.

In Figure 5 and Figure 6 we can see the segmentation mAP and loss on the validation set during training, respectively. The mAP increased in fast way until iteration 2000 when the learning rate dropped to 0.0001. After that, the mAP grows slowly until it seems to converge between iterations 3500 and 4000. On the other hand, the validation loss follows a typical behavior of decreasing along the iterations until it seems to converge since iteration 3500. Also, we can note that the decay of the learning rate helps the loss to converge. This can be seen before iteration 1000 where the loss gets stuck and begins to increase until the learning rate drops to 0.001 and the loss begins to decrease again.

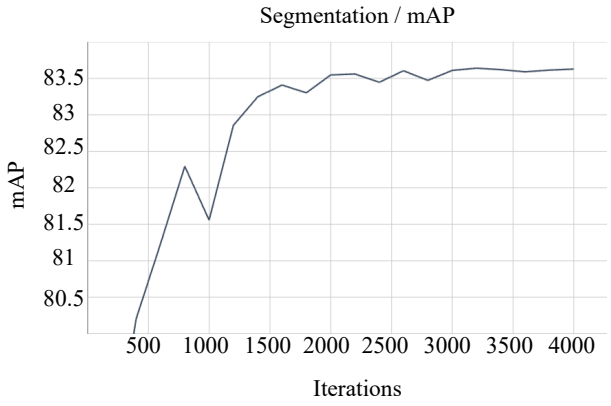

Figure 5: Evolution of the mAP on the validation set during the training.

Figure 6: Evolution of the validation loss during the training.

B. Testing and evaluation of the optimized Mask R-CNN

The Mask R-CNN model obtained after the traning is good in terms of instances segmentation mAP but not in the number of frames per second (FPS) that it can process (4 FPS) nor in the size of the model (335 MB). Therefore, if it is desired to use it in real-time systems, it will be inefficient and would need high computing resources. For this reason, it is necessary to optimize the model to process more FPS with minimal GPU power and lose minimal mAP enabling running it even in embedded systems where resources like the GPU capacity and memory space are limited. Therefore, three quantization methods of optimization are evaluated over the test images of the StrawDI Db1 dataset: FP32, FP16, and INT8.

The optimization is performed using the NVIDIA TensorRT SDK. The main criterion used to evaluate the models is the mAP. In addition, both, the FPS that the model can process and the total amount of memory that the model may occupy, are important factors to take into account at the moment to decide which is a better model for our purpose. The best model will be the one that can perform instance segmentation with high mAP and a great amount of FPS without demanding high memory consumption. This is the ideal and desired balance between the three metrics to be feasible to be used in real-time applications without converting the GPU capacity into an influencing factor. It is important to say that after using the TensorRT optimized models we noted that the GPU usage is minimal.

A comparison of the metrics obtained by the non-optimized Mask-RCNN described above, the models produced by the different quantization methods, and the ones obtained by Borrero *et al.* [10] are described in Table II. Note that the implementation of the optimization in the FP16 and INT8 models produces a minimal reduction of the mAP while the FPS gaining and the reduction of the size of the model are significant. The reduction of the mAP value was minimal, close to 1% less, compared to other approaches. Therefore, optimization with NVIDIA TensorRT seems to be a good way to optimize AI models for real-time applications. This may be due to the leadership of NVIDIA in the Deep Learning and AI fields of research. They have managed to develop efficient optimization methods which translate into getting the best performance in Deep Learning models.

Table II: Mask RCNN results on the test set for segmentation after the optimization phase according to the mAP.

Metrics	Models				
	Detectron2	FP32	FP16	INT8	Borrero *et al.* [10]
mAP	83.32	82.94	**82.93**	**82.84**	43.85
mAP_{50}	96.60	96.58	96.62	96.52	74.24
mAP_{75}	90.50	90.33	90.41	90.46	45.13
mAP_{small}	50.93	50.77	50.69	50.67	07.54
mAP_{medium}	88.12	87.34	87.41	87	51.77
mAP_{large}	96.86	97.53	97.41	97.42	75.90
FPS	4	6.15	20.3	**21.24**	10
MB	335	220	88	**48**	-

From the models displayed in Table II, we have two main options that satisfy most of the parameters: INT8 and FP16 models. On the one hand, if the user wants a lighter model with a higher amount of FPS, the best option is the INT8 model. On the other hand, if the user wants a model with a better performance in terms of mAP, the best option is the FP16 model. As the main purpose of our work is to present an efficient strawberry segmentation model, we should take into account the one that satisfies most of the metrics. Therefore, the model that fulfills our requirements is the TensorRT INT8 quantization model because it has been shown to be efficient

in terms of size and FPS (48 MB and 21.24 FPS respectively) preserving a high mAP value which is 82.84, only 0.48 down by the reference model training in Detectron2 which is about 335 MB and only process 4 FPS.

Figure 7 shows the segmentation masks applied by our model over the INT8 quantization method. The results show that the model is capable to assign segmentation masks to the strawberry fruits with high precision. However, the mask is assigned with less detail than the Mask R-CNN model trained in Detectron2. This is evidenced by observing the contour of the strawberries, which looks irregular unlike the masks assigned by the non-optimized model, shown in Figure 4.

Figure 7: Examples of segmentation performed by the Mask R-CNN model with INT8 quantization method.

Finally, a comparison between the images with non-accurate segmentations from Borrero *et al.* [10], and those obtained with our best-optimized model are shown in Figure 8. These images include strawberries with similar colors to the leaves, and therefore the segmentation in the benchmark model is not accurate. However, it is visually evidenced that our model performs better despite these conditions. Therefore, our TensorRT INT8 optimized model beats the proposed optimized model presented by Borrero *et al.* [10] in terms of mAP and FPS.

V. Conclusions and Future Work

In this study, an optimized instance segmentation model for strawberry segmentation using the Mask-RCNN architecture has been presented. The model was trained using the StrawberryDI dataset, and the optimization process was carried out using the TensorRT SDK. Our objective was to obtain a model suitable for use in real-time systems, without significantly reducing its performance on the segmentation task. For this

(a) Model proposed in [10] (b) INT8 Model

Figure 8: Comparison of segmentation performed by the model proposed in [10] (left) and the optimized INT8 model (right).

reason, not only the mAP metric is relevant, but also the FPS and the size of the model must be taken into account.

After the optimization process, three models were obtained: FP32, FP16, and INT8. These models presented a minimal mAP loss, reaching up to 1% less in the INT8 model compared to the non-optimized model. The FP32 model obtained an mAP of 82.94%, the highest among the three models. However, its performance in FPS was the lowest, obtaining only 4 on this metric which does not meet the objective of the study. In the case of the FP16 and INT8 models, the first obtained higher mAP (82.93% vs 82.84%), but the second achieved the highest value in terms of FPS (21.24 FPS vs 20.3 FPS). Although the difference in these metrics is minimal, the size of the INT8 model is almost half of the FP16 (48 MB vs 88 MB). This is relevant when implementing these models in real-time systems. Usually, the model must have to share resources with other processes, and also, the available computational resources could be limited. Therefore, the INT8 is the model that meets the mAP, FPS, and MB requirements and was selected among all the obtained models, as the most efficient for the strawberry instance segmentation task.

Future work could focus on the deployment and testing of our model in embedded systems. Furthermore, although our model presents a good performance in the segmentation task, since the StrawDI dataset includes only the strawberry class, it is also limited to a single class. Since the dataset images include strawberries with a great diversity of characteristics, it would be interesting to label the data set according to the fruit

maturity level. In this manner, the model could be retrained using transfer learning in order to obtain a model capable of not only segmenting strawberries but also identifying the least and most ripe fruits. Finally, another task to could be addressed is to compare the performance of our model with other architectures such as the recently released YOLOv7, which supports instance segmentation and promises high performance in terms of mAP and FPS.

REFERENCES

[1] H. Kang and C. Chen, "Fruit detection and segmentation for apple harvesting using visual sensor in orchards," *Sensors*, vol. 19, no. 20, p. 4599, 2019.

[2] Z. Song, Z. Zhou, W. Wang, F. Gao, L. Fu, R. Li, and Y. Cui, "Canopy segmentation and wire reconstruction for kiwifruit robotic harvesting," *Computers and Electronics in Agriculture*, vol. 181, p. 105933, 2021.

[3] U. Afzaal, B. Bhattarai, Y. R. Pandeya, and J. Lee, "An instance segmentation model for strawberry diseases based on mask r-cnn," *Sensors*, vol. 21, no. 19, p. 6565, 2021.

[4] Y. Yu, K. Zhang, L. Yang, and D. Zhang, "Fruit detection for strawberry harvesting robot in non-structural environment based on mask-rcnn," *Computers and Electronics in Agriculture*, vol. 163, p. 104846, 2019.

[5] F. Crespo, A. Crespo, L. M. Sierra-Martínez, D. H. Peluffo-Ordóñez, and M. E. Morocho-Cayamcela, "A computer vision model to identify the incorrect use of face masks for covid-19 awareness," *Applied Sciences*, vol. 12, no. 14, p. 6924, 2022.

[6] M. Halstead, C. McCool, S. Denman, T. Perez, and C. Fookes, "Fruit quantity and ripeness estimation using a robotic vision system," *IEEE robotics and automation LETTERS*, vol. 3, no. 4, pp. 2995–3002, 2018.

[7] Z. Al-Mashhadani and B. Chandrasekaran, "Autonomous ripeness detection using image processing for an agricultural robotic system," in *2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. IEEE, 2020, pp. 0743–0748.

[8] D. Mao, H. Sun, X. Li, X. Yu, J. Wu, and Q. Zhang, "Real-time fruit detection using deep neural networks on cpu (rtfd): An edge ai application," *Computers and Electronics in Agriculture*, vol. 204, p. 107517, 2023.

[9] L. Zhang, G. Gui, A. M. Khattak, M. Wang, W. Gao, and J. Jia, "Multi-task cascaded convolutional networks based intelligent fruit detection for designing automated robot," *IEEE Access*, vol. 7, pp. 56 028–56 038, 2019.

[10] I. Perez-Borrero, D. Marin-Santos, M. E. Gegundez-Arias, and E. Cortes-Ancos, "A fast and accurate deep learning method for strawberry instance segmentation," *Computers and Electronics in Agriculture*, vol. 178, p. 105736, 2020.

[11] M. A. Fadhel, A. S. Hatem, M. A. E. Alkhalisy, F. H. Awad, and L. Alzubaidi, "Recognition of the unripe strawberry by using color segmentation techniques," *International Journal of Engineering and Technology (UAE)*, vol. 7, no. 4, pp. 3383–3387, 2018.

[12] A. Durand-Petiteville, S. Vougioukas, and D. C. Slaughter, "Real-time segmentation of strawberry flesh and calyx from images of singulated strawberries during postharvest processing," *Computers and electronics in agriculture*, vol. 142, pp. 298–313, 2017.

[13] C. Ouyang, D. Li, J. Wang, S. Wang, and Y. Han, "The research of the strawberry disease identification based on image processing and pattern recognition," in *Computer and Computing Technologies in Agriculture VI: 6th IFIP WG 5.14 International Conference, CCTA 2012, Zhangjiajie, China, October 19-21, 2012, Revised Selected Papers, Part I 6*. Springer, 2013, pp. 69–77.

[14] T. Wang, K. Zhang, W. Zhang, R. Wang, S. Wan, Y. Rao, Z. Jiang, and L. Gu, "Tea picking point detection and location based on mask-rcnn," *Information Processing in Agriculture*, 2021.

[15] L. Zu, Y. Zhao, J. Liu, F. Su, Y. Zhang, and P. Liu, "Detection and segmentation of mature green tomatoes based on mask r-cnn with automatic image acquisition approach," *Sensors*, vol. 21, no. 23, p. 7842, 2021.

[16] L. Costa, Y. Ampatzidis, C. Rohla, N. Maness, B. Cheary, and L. Zhang, "Measuring pecan nut growth utilizing machine vision and deep learning for the better understanding of the fruit growth curve," *Computers and Electronics in Agriculture*, vol. 181, p. 105964, 2021.

[17] J. Lee, H. Nazki, J. Baek, Y. Hong, and M. Lee, "Artificial intelligence approach for tomato detection and mass estimation in precision agriculture," *Sustainability*, vol. 12, no. 21, p. 9138, 2020.

[18] W. Jia, Y. Tian, R. Luo, Z. Zhang, J. Lian, and Y. Zheng, "Detection and segmentation of overlapped fruits based on optimized mask r-cnn application in apple harvesting robot," *Computers and Electronics in Agriculture*, vol. 172, p. 105380, 2020.

[19] Y. Ge, Y. Xiong, and P. J. From, "Instance segmentation and localization of strawberries in farm conditions for automatic fruit harvesting," *IFAC-PapersOnLine*, vol. 52, no. 30, pp. 294–299, 2019.

[20] Y. Yu, K. Zhang, L. Yang, and D. Zhang, "Fruit detection for strawberry harvesting robot in non-structural environment based on mask-rcnn," *Computers and Electronics in Agriculture*, vol. 163, p. 104846, 2019.

[21] X. Wang, X. Yue, H. Li, and L. Meng, "A high-efficiency dirty-egg detection system based on yolov4 and tensorrt," in *2021 International Conference on Advanced Mechatronic Systems (ICAMechS)*, 2021, pp. 75–80.

[22] L. Tao, T. Hong, Y. Guo, H. Chen, and J. Zhang, "Drone identification based on centernet-tensorrt," in *2020 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, 2020, pp. 1–5.

[23] Y. Lan, K. Huang, C. Yang, L. Lei, J. Ye, J. Zhang, W. Zeng, Y. Zhang, and J. Deng, "Real-time identification of rice weeds by uav low-altitude remote sensing based on improved semantic segmentation model," *Remote Sensing*, vol. 13, no. 21, p. 4370, 2021.

[24] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[25] R. Mask, K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer*, 2017, pp. 2961–2969.

[26] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[27] NVIDIA. (2023) NVIDIA TensorRT. [Online]. Available: https://developer.nvidia.com/tensorrt