IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 8, 2020, accepted December 26, 2020, date of publication December 31, 2020,
date of current version January 11, 2021.

Digital Object Identifier 10.1109/ACCESS.2020.3048374

End-to-End Automatic Berry Counting for Table Grape Thinning

PRAWIT BUAYAI[1], KANDA RUNAPONGSA SAIKAEW[2], AND XIAOYANG MAO[1,3], (Member, IEEE)

[1]Faculty of Engineering, University of Yamanashi, Kofu 400-0015, Japan
[2]Faculty of Engineering, Khon Kaen University, Khon Kaen 40002, Thailand
[3]School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou 310013, China

Corresponding author: Xiaoyang Mao (mao@yamanashi.ac.jp)

ABSTRACT Bunch shape and berry size indicate the quality of table grapes and crucially affect their market value. Berry thinning is one of the most important tasks in grape cultivation to achieve an ideal bunch shape and to make sufficient space for individual berries. A successful practice by skilled grape farmers in Japan is using the number of berries in a bunch to guide the thinning process; hence, a technique for automatically counting the number of berries in a working bunch has been long desired by farmers to improve the efficiency of the thinning task. This research presents a novel end-to-end berry-counting technique based on a deep neural network (DNN), and its contributions are as follows. First, because a DNN requires massive training data, a novel data augmentation technique simulating the thinning process is proposed. Second, a new location-sensitive object detection model that integrates explicit location information and supplementary classification loss into a state-of-the-art instance segmentation model was proposed for detecting the number of berries in a working bunch with a high accuracy. Third, a set of features, together with their extraction algorithms, is designed for predicting the number of berries in a bunch (3D counting) using the berries detected on a single 2D image. Experiments using data collected from farmers' grape-thinning process have been conducted to validate the accuracy and effectiveness of the proposed methods.

INDEX TERMS Smart agriculture, grape detection, berry thinning, berry counting, instance segmentation, deep neural network.

I. INTRODUCTION

Table grape production requires high-quality grapes, and essential factors include the bunch compactness, bunch shape, and berry size [1], [2]. To produce high-quality table grapes, a critical process called berry thinning is necessary to remove unnecessary berries. Berry thinning benefits not only table grape but also wine grape production. Karoglan *et al.* [3] found that a combination of bunch thinning and berry thinning reduced the grape yield but increased the mean cluster weight, total phenols, flavan-3-ols, and anthocyanins, as well as many individual phenolic compounds. Likewise, the grape bunch becomes more open and less inclined to disease development [2], [4], [5].

Fig. 1 shows a bunch during the thinning process. After thinning, the bunch should have a compact and well-balanced shape, and each berry should have sufficient space to grow to the desired size without interfering with others. A successful practice by skilled grape farmers in Japan for achieving such a requirement is using the number of berries in the working

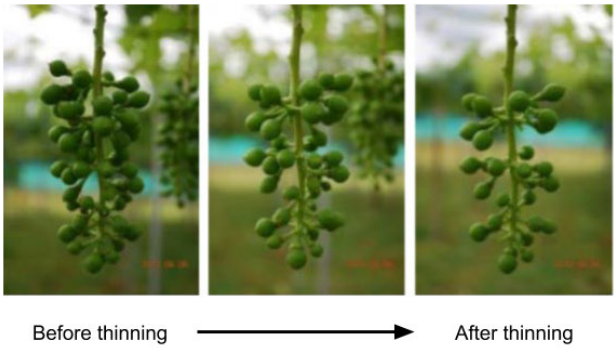

FIGURE 1. The berry-thinning process. Before thinning, the bunch was crowded with berries. After thinning, the bunch had a fine shape and a lesser likelihood of berry decay.

The associate editor coordinating the review of this manuscript and approving it for publication was Tallha Akram.

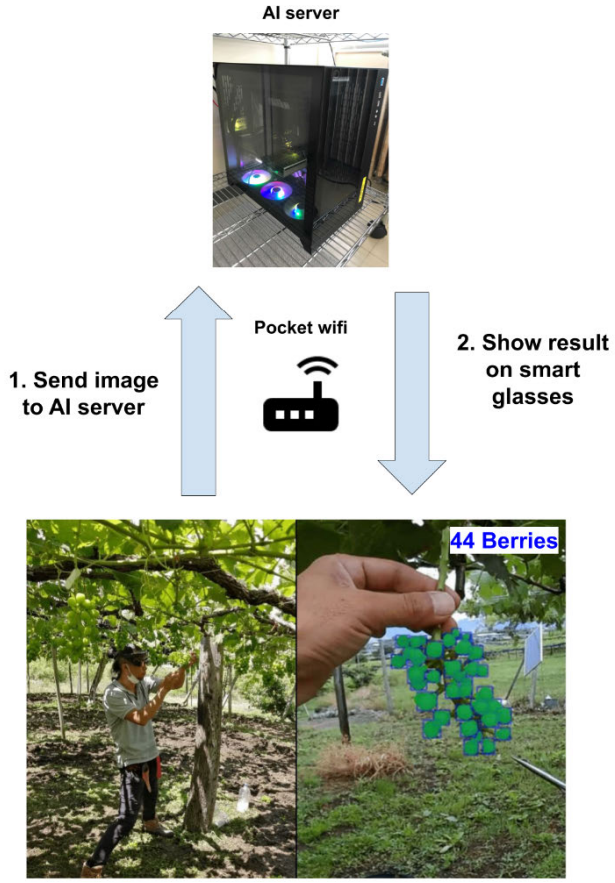

FIGURE 2. Applying the proposed end-to-end automatic berry-counting technique to table grape thinning. Smart glasses send the captured image to the artificial intelligence (AI) server via pocket WiFi. The AI server then sends the result to show the detected berries and estimate the number of berries using 3D counting (including hidden berries).

bunch to guide the thinning process. Given the desired overall shape of the bunch and the full size of grown berries, the number of berries in the working bunch is a good indicator of whether sufficient space has been created through thinning. The ideal range of berry numbers for typical table grape varieties in Japan is shown in Table 1. However, counting berries during berry thinning is time-consuming and is especially difficult for inexperienced farmers. Furthermore, the suitable period for berry thinning is limited to one to two weeks, when there is still enough space among berries to allow unnecessary berries to be cut without hurting those to be kept, before the grapes start to accumulate sugar [6]. For the above reasons, an automatic berry-counting technology is desired by grape farmers.

In this paper, we propose a novel end-to-end automatic berry-counting technology for supporting the berry-thinning process. Fig. 2 depicts the application of the proposed technology in real berry-thinning tasks. Smart glasses (Epson Moverio BT-2000 in current implementation) are used to capture images and show the farmers the predicted berry numbers in a single working bunch. Smart glasses make it possible to avoid interrupting farmers' normal tasks. The server-based

TABLE 1. The expected number of berries in the bunch according to grape variety [7].

Grape variety	Expected number of berries
Fujiminori	28–30
Pione	32
Black beet	32
Kyoho	35–40

approach is adopted to make use of state-of-the-art-of deep neural network (DNN) models. The technical contributions of this paper can be summarized as follows:

1. A novel data augmentation technique that can automatically generate training datasets that simulate the berry-thinning process. Because berry thinning is conducted once a year during a short period, collecting a large training dataset corresponding to different weather and location conditions is highly difficult but extremely important. The proposed method makes it possible to generate automatically a large annotated training dataset from a small dataset.
2. A novel location-sensitive object detection model, realized as an extension of the state-of-the-art instance segmentation DNN model, to detect the berries in a working bunch only. Location invariant is a property of DNN models inherently realized through the pooling layers, making it possible to detect all objects with the learned features regardless of their locations in the images. Such a property, however, is undesirable for our berry-thinning support purpose, as it means the berries of not only the working bunch but all bunches in an image will be detected. We solved the problem by integrating location information into the Hybrid Task Cascade (HTC) mask R-CNN model [33].
3. A novel method to estimate the number berries in a bunch from one single 2D image of the bunch. Because grape berries have a round shape and no distinguishing features that can be tracked individually, it is difficult and computationally expensive to track and count all individual berries. Our method succeeded in achieving a high prediction accuracy that can withstand practical use via a set of originally designed features detected from single 2D images.

The rest of this paper is organized as follows. Section 2 describes related works. Section 3 introduces the details of the proposed method. Section 4 presents and discusses the experimental results. Finally, Section 5 offers our conclusions and future research directions.

II. RELATED WORKS

Smart agriculture is now gaining significant attention, and the use of computer vision is crucial for various applications. Yield prediction is one of the most important applications. Technologies for automatically detecting various fruits, such as grapes [1], [4], [16]–[20], [8]–[15], oranges [21],

apples [21], tomatoes [22], sweet peppers [23], and cherries [24], have been developed. Many parts of a grape can be detected and used to estimate the yield in a vineyard, including the berry [1], [15], [16], [18], [19], [25]–[27], flower [11], [20], bunch [4], [8], [12], [17], and shoot [14]. In the remainder of this section, we refer to these existing methods from two perspectives: berry detection and berry counting.

A. BERRY DETECTION

Considering the round shape of the grape, the circle Hough Transform (CHT) has been employed to detect grape berries. Roscher *et al.* [25] introduced the CHT to detect grape berries in the natural scene, while Liu *et al.* [10] employed the CHT for preprocessing in the 3D reconstruction of a grape bunch from a single image. In addition, Rudolph at el. [20] applied the CHT during post-processing to filter the flower bunch detected from the DNN network. However, a major problem of the CHT-based approach is that it cannot detect berries partially occluded by other berries. Reis *et al.* [8] and Luo *et al.* [12] proposed a system for detecting grape bunches in the natural environment based on the color mapping approach. Aquino *et al.* [15], [16], [11], [28] proposed a method for estimating the number of grapevine berries and flowers using image analysis based on the h-maxima transform. Nuske*et al.* [26] and Pérez-Zavala *et al.* [17] use feature descriptors, such as histograms of gradients (HoGs), fast retina keypoint (FREAK), local binary patterns (LBPs), and scale-invariant feature transform (SIFT), to detect the berries. However, the above approaches may not operate in a natural field with uneven illumination conditions and shadows. Such a problem can be solved with a DNN, because the image feature can be trained in the model, not just by using the specific range of the color value or specific hand-craft features to distinguish the objects [4], [19]. Typically, the approaches based on semantic segmentation are not designed to count object instances in the image, as the result of semantic segmentation is pixel-wise and overlapping objects of the same class cannot be distinguished. Zabawa *et al.* [19] tried to solve such a problem when applying semantic segmentation to grape berry detection by introducing a new edge class object surrounding the individual grape berry. However, because the edge is a small object, it is difficult to detect all edge pixels surrounding the berry. The method based on instance segmentation was designed to give an output comprised of the bounding box, classification, and pixel mask, thus immediately counting the individual objects. Santos *et al.* [4] used instance segmentation to detect a grape bunch, but the detection of berries was not addressed in their study. Most importantly, none of the recent DNN-based approaches [4], [18], [19] introduced a method for focusing on a particular bunch, which is crucial in supporting the table grape-thinning task.

B. BERRY COUNTING

While our method can operate using a mobile device in a real field, existing research dealing with the number of berries in 3D bunches has required a laboratory setup or special capture devices. For instance, Liu *et al.* [10] required a plain background to apply Otsu's binarization while rotating the grape bunch. Ivorra *et al.* [1] needed constant light intensity, so they installed a stereo camera using four pairs of fluorescent tubes to afford the illumination. Schöler *et al.* [29] installed a laser scanner on a robot arm to scan the grape bunch. Because 3D reconstruction operation is time consuming, it is not appropriate for real grapevine yard application.

III. PROPOSED METHOD

Fig. 3 depicts the framework of the proposed end-to-end automatic berry-counting technique. The framework consists of three parts: a DNN model that takes a captured 2D image as the input and detects the berries in a working bunch, a feature extractor that computes a set of carefully designed features from the detected berries, and a regression model that predicts the number of berries in the whole 3D bunch using the features from the feature extractor. For the DNN model, we made extension for the HTC [33], a state-of-the-art instance segmentation model, to detect the berries only in the working bunch and to exclude other bunches. A new data augmentation technique is proposed to generate a large dataset to train this extended DNN model. To predict the number of berries in the whole 3D bunch, a set of features together with their extraction algorithms is carefully designed, and three different regression models are investigated. The details of each part of the framework are given in the remainder of this section.

A. DATA AUGMENTATION

Deep learning models have gained huge success in object detection tasks [30]–[33]. However, to train a successful model, a large amount of labeled data is required. Because berry thinning is performed once a year during a short period, it is difficult to collect a sufficient number of images for training a model that can accurately detect berries during the whole process of berry thinning. Moreover, for the training of an instant segmentation model, the masks of individual grape berries are required. Generating such annotated data with manual labeling requires a huge amount of labor. To solve this problem, this study proposes a new data synthesis method to generate sufficient data from a small training set. The basic idea is to generate the images simulating the thinning process by removing berries gradually from an existing image. As shown in Fig. 4, removing a front berry may result in an unnatural appearance of the berries partially occluded by this front berry. To avoid such an artifact, the proposed method first identifies the berry behind the front berry by computing the circularity of the berries. If the circularity is below a given threshold, we can judge that it is a partially occluded berry and it can be removed. To make the synthesized image look as natural as possible, a state-of-the-art image inpainting technology using a deep convolutional neural network [34] is employed to fill the region of the removed berry. Fig. 5 shows the process of our synthesis method. First, a partially occluded berry is identified by computing the circularity

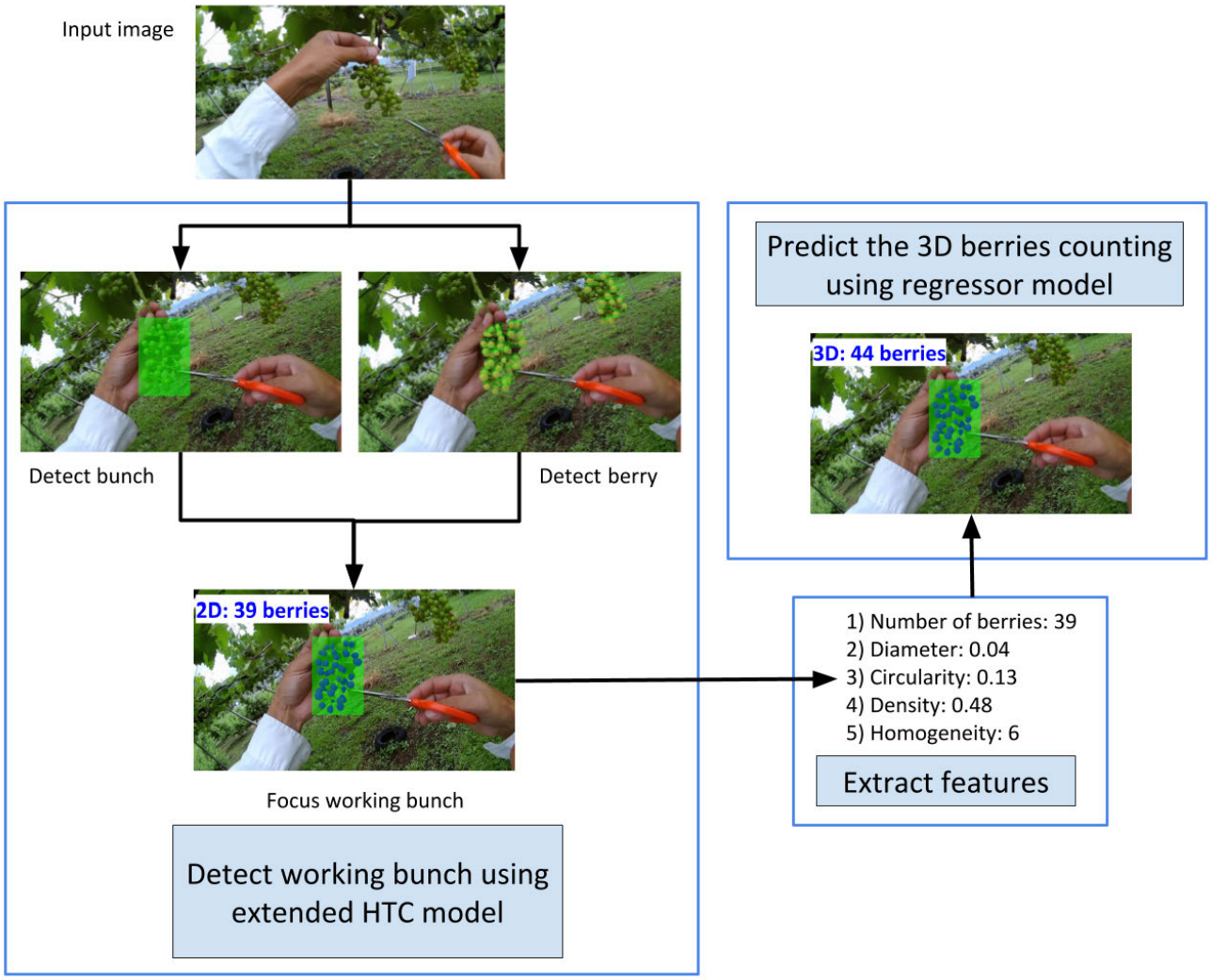

FIGURE 3. The framework of the proposed end-to-end automatic berry-counting technique for table grape thinning.

(Fig. 5 [b]); then, this berry is removed (Fig. 5 [c]); finally the removed region is filled with inpainting technology (Fig. 5 [d]). This process can be repeated until all the partially occluded berries are removed, simulating the images captured during the thinning process. Two examples of synthesized images are shown in Fig. 6.

B. AUTOMATIC FOCUSING ON WORKING BUNCH

1) LOCATION SENSITIVE HTC MODEL

As depicted in Fig. 2, this research aims to support farmers in effectively performing grape thinning by visualizing the number of berries in a working bunch. Therefore, the DNN model used for detecting berries should meet three requirements. First, it should be able to detect the berries only in the working bunch without detecting the berries in other bunches in the captured images. Second, it should detect the berries with a high accuracy without detecting the same berry multiple times. Third, as will be introduced in Part C of this section, we need the geometry features of berries to predict the number of berries in a 3D bunch; therefore, it is desirable to obtain the accurate mask of individual berries. The third requirement indicates that we should use an instance segmentation DNN model. The second requirement cannot be met by any existing DNN models, as a DNN model is actually designed to be location-invariant to detect all objects with the learned features regardless of their locations. To solve the problem, this study proposes a new location-sensitive model by integrating explicit location information into the HTC, the state-of-the-art instance segmentation model proposed by Chen *et al.* [33]. Because the location information can also be viewed as a kind of feature distinguishing the berries in the working bunch from other objects in the image, the integration of location information into the DNN model can actually improve the detection accuracy, which contributes to meeting the first requirement.

Fig. 7 depicts the network architecture of the original HTC model [33]. It consists of the CNN backbone network ('Backbone CNN') for extracting features; the region proposal

FIGURE 4. The problem occurs when synthesizing the image by removing the circular berry. The red circle is the inpainting area in which the berry was eliminated.

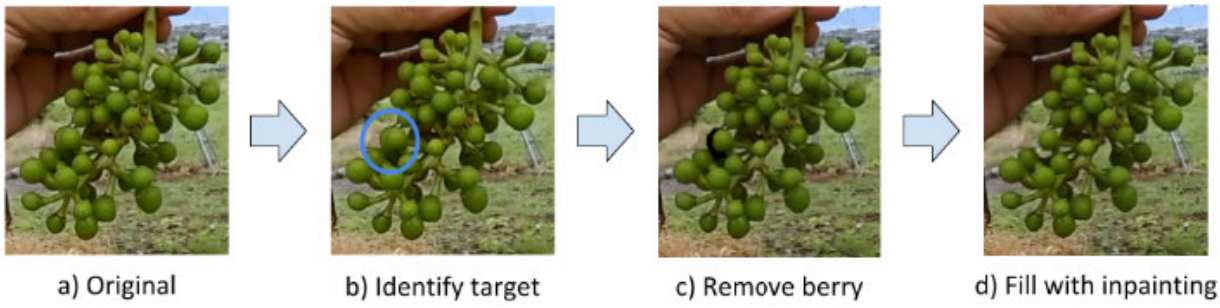

FIGURE 5. The process to synthesize new image data using the image inpainting technique.

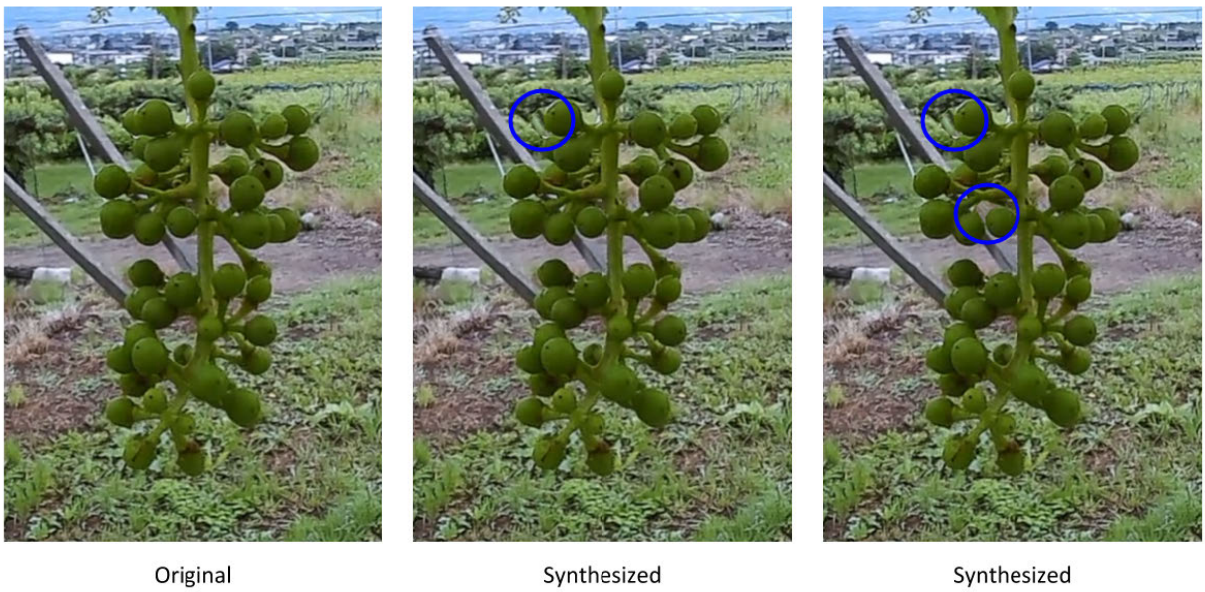

FIGURE 6. Comparison of the original image and its synthesized image. The blue circle is the inpainting area in which a berry was eliminated.

network ('RPN') for predicting the location of objects in the input image; the pooling layer ('Pooling'), which is the cropped features from the backbone network using the mapping location from the RPN; classification branches ('Class') that predict the classes of objects; bounding box branches ('BBox') that predict the locations of objects in the input

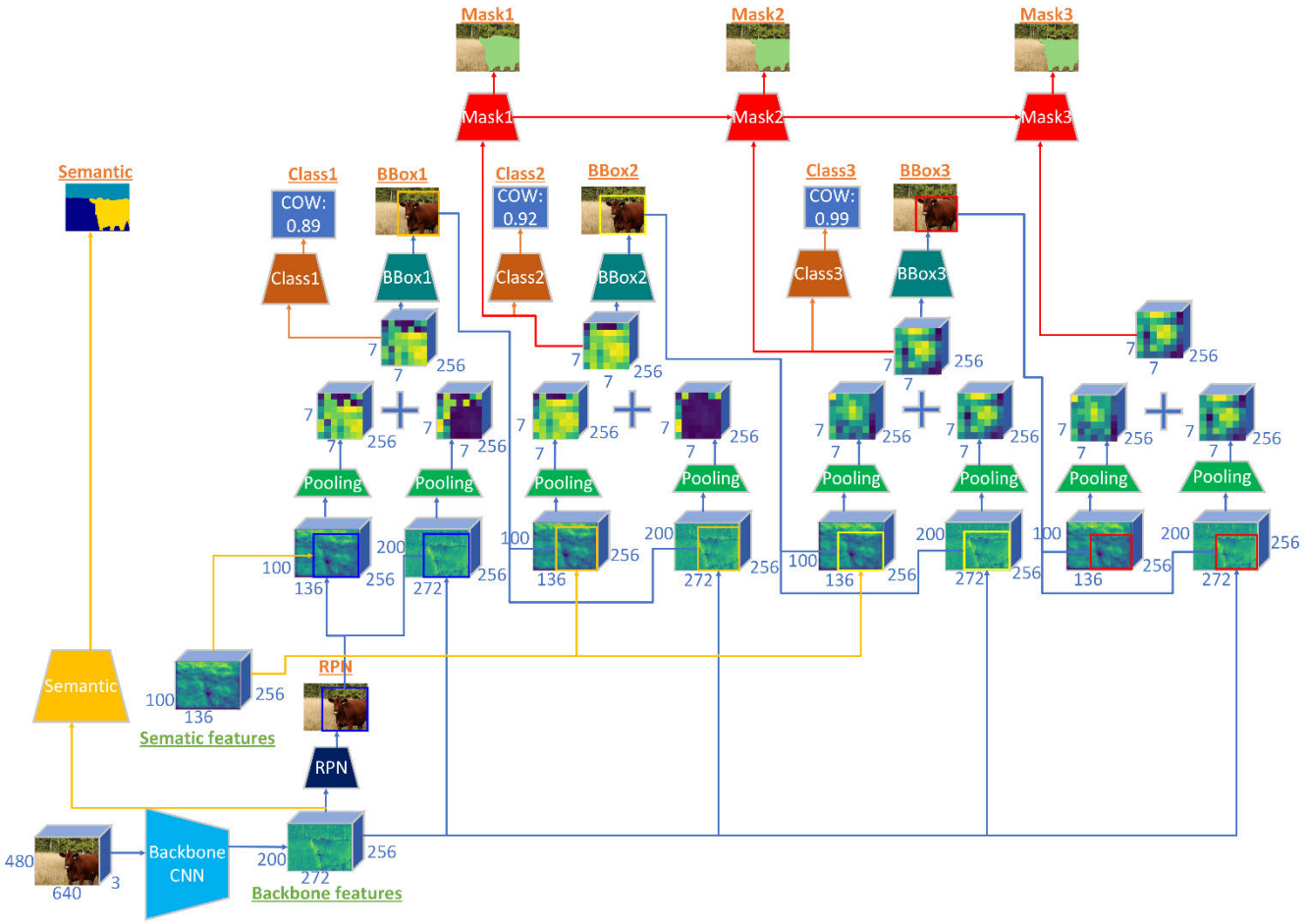

FIGURE 7. The structure of the HTC network [33].

image; mask branches ('Mask') that predict the pixel-level masks of objects; and a semantic branch ('Semantic') that predicts pixel-level stuff segmentation for the whole image.

Fig. 8 and Fig. 9 show the proposed location-sensitive HTC models with location features integrated into the Fully Connected (FC) layer and into the HTC itself, respectively. In both models, the semantic segmentation branch from the original HTC has been excluded because we only have two kinds of objects, the bunch and the berries, and we do not need stuff segmentation for the whole image. Fig. 8 shows the location feature from the RPN, BBox1, and BBox2, which were represented in terms of (x1, y1, x2, y2) and were fed as the input to the Classes and BBoxes, along with the features from the FC layer.

As shown in Fig. 9, the second method is to add the new network head, named the supplementary classification head (SCLASS), to the HTC network. We combine location features (from the RPN, BBox1, and BBox2) and the feature from the FC layer (from the pooling layer) as the input of the SCLASS branch. Because the new supplementary classification branch has been incorporated into the network architecture, defining a new supplementary loss for this branch is necessary. The HTC is a multi-stage approach; that is, at each stage t, for all sampled regions of interest (RoIs), the box branches estimate the bounding box regression offset, the classification branches estimate the classification score, and the mask branches estimate the pixel-wise masks for positive RoIs. By adding the new supplementary classification branches, the overall loss function, taking the form of multi-task learning, is defined as follows:

$$L = \sum_{t=1}^{T} \alpha_t \left(L^t_{bbox} + L^t_{cls} + L^t_{scls} + L^t_{mask} \right) \tag{1}$$

where L^t_{bbox} is the loss of the bounding box predictions at stage t and L^t_{cls} is the loss of the classification at stage t, which is the same as that of the Cascade R-CNN [32]. L^t_{scls} is the proposed loss of the classification on the new supplementary classification branch at stage t. L^t_{mask} is the loss of mask prediction at stage t, which employs binary cross entropy (BCE), as in the Mask R-CNN [31]. The coefficient α_t is used to balance the supplements of several stages and tasks. The hyper-parameter settings have been adopted from the HTC [33] with $\alpha = [1, 0.5, 0.25]$ and $T = 3$ by default.

The bounding box regression loss for each RoI in (2) is defined over a tuple of the bounding box ground truth $v = (v_x, v_y, v_w, v_h)$ and a predicted tuple $b = (b_x, b_y, b_w, b_h)$ for each class, where x, y, w, and h are the position (x, y) and size

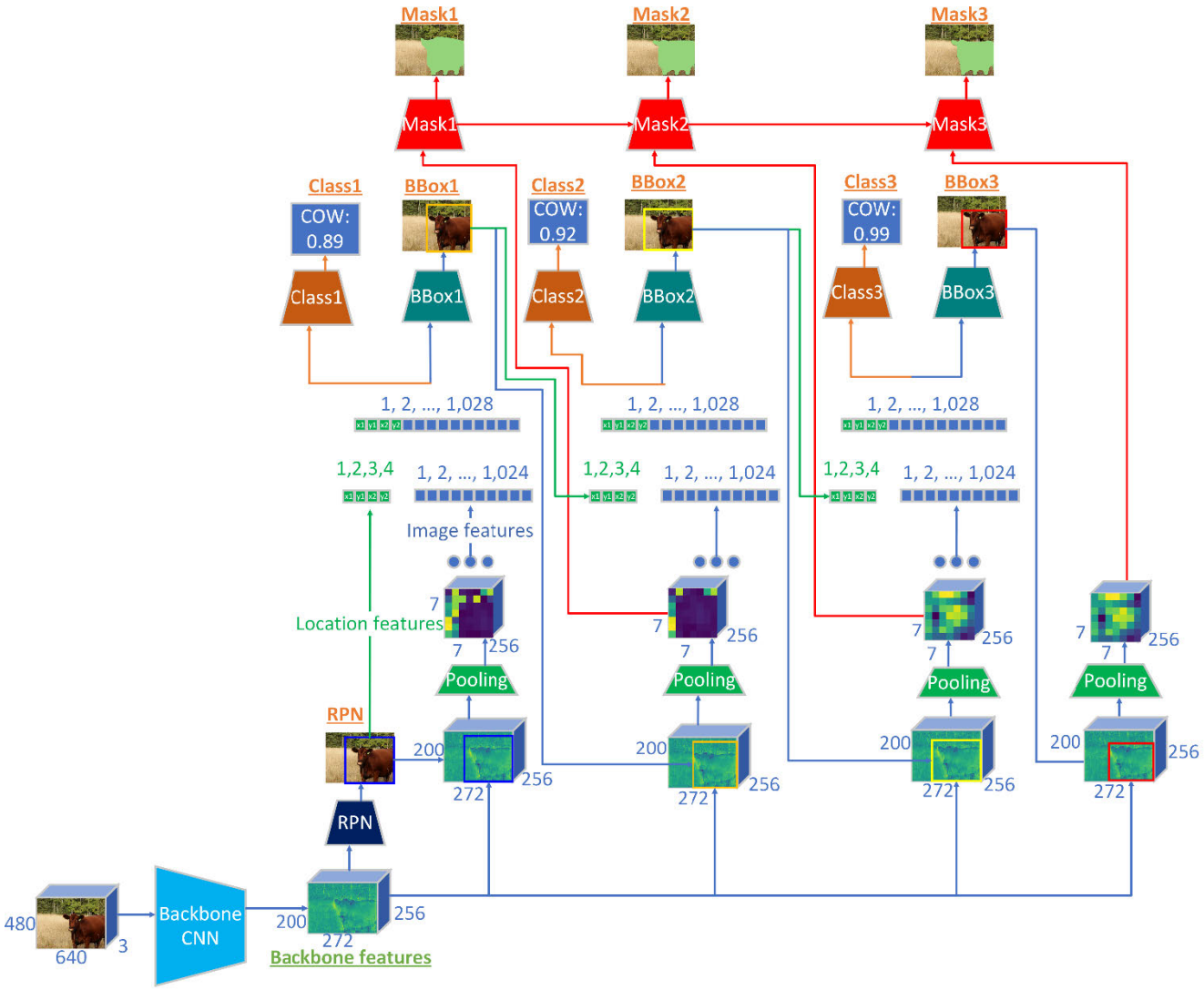

FIGURE 8. The proposed location-sensitive HTC network that integrates location features at the FC layer.

(w, h) of the RoI. L_1 is the Manhattan distance defined in (3) as in the Fast R-CNN [35].

$$L_{bbox}\,(b, v) = \sum_{i\in\{x,y,w,h\}} smooth_{L_1}\,(b_i - v_i) \tag{2}$$

$$smooth_{L_1}\,(x) = \begin{cases} 0.5x^2, & \text{if } |x| =< 1 \\ |x| - 0.5, & \text{otherwise} \end{cases} \tag{3}$$

The classification and supplementary classification loss are defined by cross entropy (CE), where p is the predicted probability computed by a softmax at the FC layer, while u is the ground truth for each class. CE loss measures the performance of a classification model whose output is a probability value between 0 and 1. CE loss increases as the predicted probability diverges from the actual label. A perfect model would have a CE loss of 0, where CE is defined as follows:

$$CE(p, u) = -\sum_{i}^{K} u_i \log p_i \tag{4}$$

where K is the number of classes in the model. In BCE loss, where the number of classes K equals 2, CE can be calculated as:

$$BCE(p, u) = -(u\log p + (1-u)\log(1-p)) \tag{5}$$

The mask branch has a Km^2 dimensional output for each RoI, which encodes the K binary masks of resolution $m \times m$, one for each of the K classes. He *et al.* [31] applied a per-pixel sigmoid and defined L_{mask} as the average BCE loss:

$$L_{mask}\left(m_{pred}, m_{gt}\right) = BCE(m_{pred}, m_{gt}) \tag{6}$$

For a RoI, m_{pred} is a predicted mask and m_{gt} is a ground-truth class u.

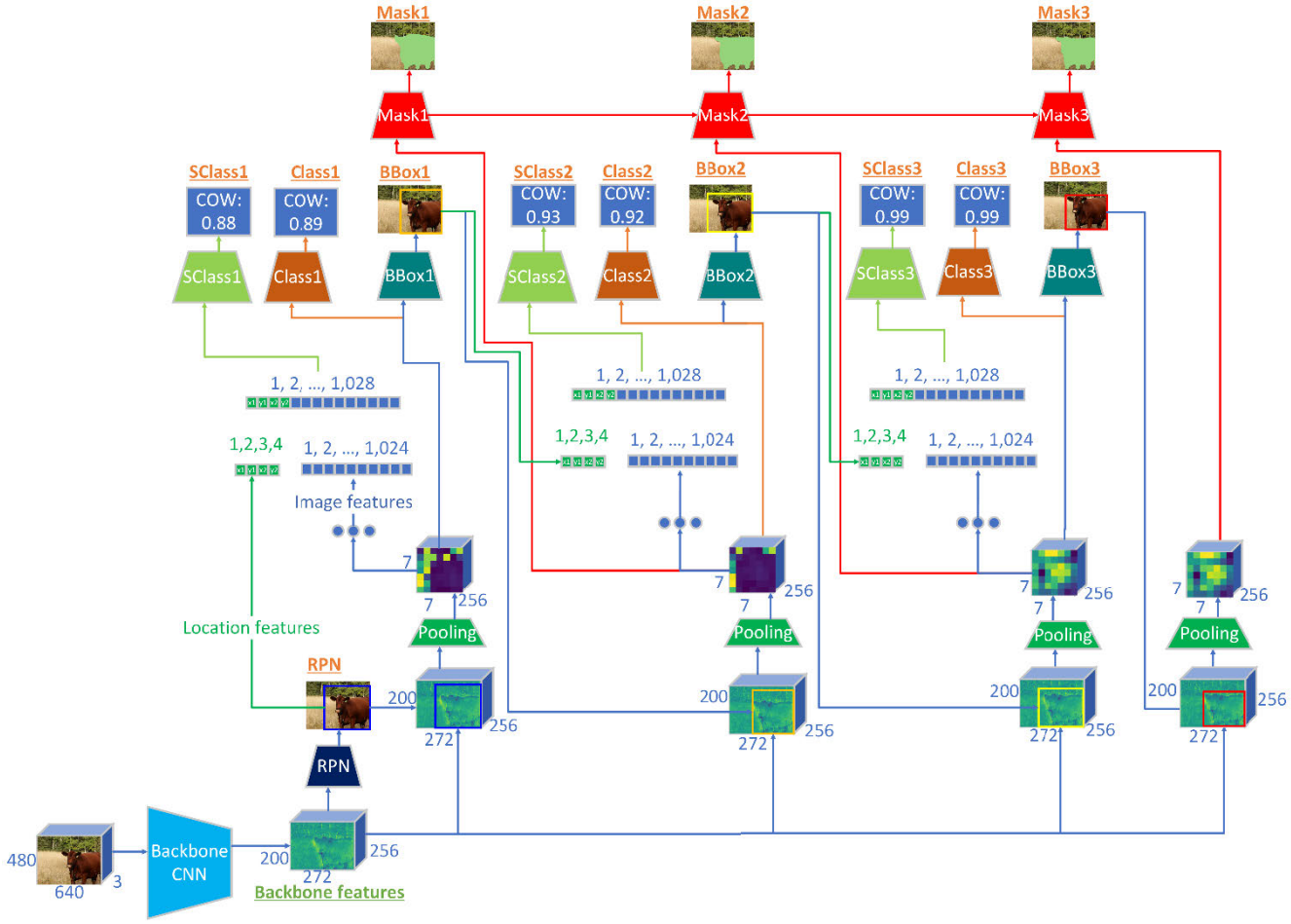

FIGURE 9. The proposed location-sensitive HTC network, which has a new 'supplementary classification branch' (SCLASS) taking the location feature and the feature from the FC Layer as the inputs.

FIGURE 10. An example in which the proposed location-sensitive HTC still detected other bunches in addition to the working bunch.

2) POST-PROCESSING

With the extended HTC models, we can detect the working bunch in most cases, but occasionally, bunches other than the working bunch may be detected. Fig. 10 shows an example in which three bunches have been detected. To exclude these bunches further, we propose post-filtering the bunches using the probability of estimation and the size of the bounding box obtained from the proposed location-sensitive HTC model. The post-processing procedure is depicted in Fig. 12. First, we remove the bunch with a low probability of estimation, and then we sort the sizes of the bunch bounding boxes to assign the bigger bounding box priority. Afterward, we adopt an intersection over union (IoU) to find the overlap bounding box. Finally, we remove the smaller overlap bounding box and select the bounding box nearest the image's center. Fig. 11 shows the result by applying the proposed post-processing technique to the results shown in Fig. 10.

C. AUTOMATIC BERRY NUMBER PREDICTION USING A SINGLE IMAGE

Predicting the number of berries in a whole 3D bunch using a single image is highly challenging, as the number of visible berries can differ significantly depending on the view directions. Based on careful observation, we found that the relationship between the number of berries in the whole 3D bunch and the number of berries visible in the captured images could be affected by multiple factors. We empirically use the

FIGURE 11. An example in which only the working bunch is detected by applying the proposed location-sensitive HTC and post-processing.

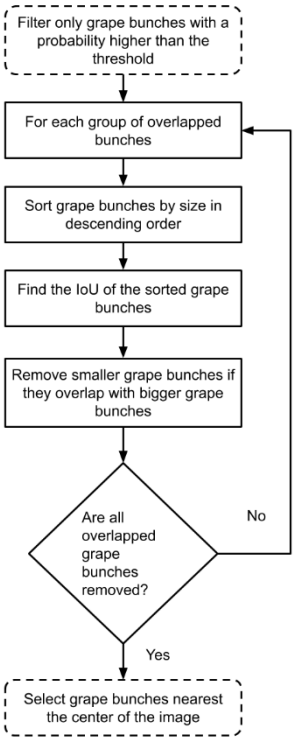

FIGURE 12. Flowchart of post-processing filtering for the further elimination of berries not included in the working bunch.

following five features computed from the 2D images as the inputs to the regression model for predicting the number of berries in a 3D bunch.

1. Number of berries
2. Diameters of berries
3. Circularity of berries
4. Density of berries
5. Homogeneity of berry distribution

The berry number feature $Feat_{nberries}$ is set to the number of detected berries (N_{cd}) in the single image as follows.

$$Feat_{nberries} = N_{cd} \tag{7}$$

The diameter feature $Feat_{diameter}$ can be computed as the average diameter of all berries detected in the 2D image with (8).

$$Feat_{diameter} = \frac{\sum_{i=1}^{N_{cd}} Berry_{diameter}(i)}{\sum_{i=1}^{N_{cd}} Berry_{area}(i)} \tag{8}$$

Here, $Berry_{diameter}$ is the diameter of individual berries. The distance between the camera and the grape in each image is not fixed, and the absolute diameter value changes with the distance. We make the feature scale invariant by normalizing the diameter with the berry area denoted as $Berry_{area}$.

The circularity feature ($Feat_{circularity}$) indicates how many partially occluded berries are among the detected berries. Generally, the occluded berries have a non-circular shape. The circularity of a berry [36] can be computed with (9) from the berry's area $Berry_{area}$ and perimeter $Berry_{perimeter}$.

$$Berry_{circularity} = \frac{4\pi Berry_{area}}{Berry_{perimeter}^2} \tag{9}$$

The circularity feature $Feat_{circularity}$ is then computed as the proportion of the number of occluded berries ($Berry_{circularity}$ less than the threshold) over the total number of detected berries (N_{cd}) with (10).

$$Feat_{circularity} = \frac{\sum_{i=1}^{N_{cd}} \begin{cases} 1, & \text{if } Berry_{circularity}(i) \leq 0.7 \\ 0, & \text{otherwise} \end{cases}}{N_{cd}} \tag{10}$$

The detected non-occluded berries should have a round shape with a circularity value close to 1.0. The number of partially occluded berries can be estimated by counting the number of berries whose circularity is smaller than a given threshold, which is empirically set to 0.7 in our experiment.

The density feature $Feat_{density}$ is computed with (11) as the proportion of the berries' area $Berries_{area}$, which is the summation of the areas of all detected berries, over the bunch area $Bunch_{area}$, as detected by the location-sensitive HTC model trained with bunch images.

$$Feat_{density} = \frac{\sum_{i=1}^{N_{cd}} Berries_{area}(i)}{Bunch_{area}} \tag{11}$$

The larger the $Feat_{density}$, the more berries are likely to be occluded in the current image. Therefore, $Feat_{density}$ also gives a reasonable indication of the number of occluded berries.

The homogeneity feature $Feat_{homo}$ indicates how uniform the distribution of the detected berries is in the image. We found that the distribution of berries can be non-uniform in the images, which means severe occlusion can occur locally even though the overall density is low. Therefore, together with the density, the homogeneity feature also plays an important role in accurately predicting the number of berries in a 3D bunch. To compute the homogeneity feature, we employ a method based on Gaussian smoothing [37]. We consider the mask image with the berry area set to white

(255) and the other set to black (0). If the berries are uniformly distributed, that is, if each berry is surrounded by the background and no berries are overlapping or close to each other, then after repeatedly applying Gaussian smoothing, the berry area will be gradually blended with the background. Thus, we can obtain an image of uniformly gray pixel values. On the contrary, if the berries are not uniformly distributed, then the image should consist of a large background area and an area with dense overlapping berries. Then, some background areas and berry areas would remain unchanged, even after repeatedly applying Gaussian smoothing. Therefore, the difference between the images at different stages of repeated Gaussian smoothing should give a good measure of the homogeneity. In our current implementation, we compute the difference between the images after applying Gaussian smoothing once and the image after applying Gaussian smoothing 11 times and then adding the difference of all pixels together to get the $Feat_{homo}$.

To predict the number of berries in a 3D bunch using the above five features, we experimented with six representative regression models: kernel ridge regression (KRR) [38], support vector regression (SVR) [39], random forest regression (RFR) [40], gradient boosting (GB) [41], stochastic gradient descent (SGD) [42], and artificial neural network (ANN) [43]. The results are presented in Section IV.E.

IV. EXPERIMENT RESULTS AND DISCUSSION

A. EXPERIMENT SETTING

1) DATASET AND IMPLEMENTATION DETAILS

We asked two farmers for help by installing cameras on their heads to capture the working scene during the grape-thinning task. Then, we manually labeled 2,701 berries in 60 images from 10 different bunches. Each image has a resolution of 1,920 × 1,080 pixels, and each was rescaled to have a minimum size of 800 pixels and a maximum size of 1,333 pixels. In the current implementation, the models are trained and evaluated on a single Titan RTX GPU for more than 10 hours.

The hyper-parameter settings followed those of the HTC [33]. Stochastic gradient descent [44] was employed to train the model. The weight decay and momentum are set to 0.0001 and 0.9, respectively, and the batch size is set to 1. According to the Linear Scaling Rule [45], the learning rate is set to 0.00125.

2) EVALUATION METRICS

Because the aim of the proposed technique is to detect grape berries accurately, we measure the accuracy by computing the IoU between the mask of the detected grape berry and that of the ground truth grape berry. Similar to [19], we use two quantitative measures, Correctly Detect (*CD*) and Miss-Classification (*MC*), which are computed with (12) and (13), respectively.

$$CD = \left(\frac{N_{cd}}{N_{gt}}\right) \times 100 \tag{12}$$

$$MC = \left(\frac{N_{id}}{N_{ad}}\right) \times 100 \tag{13}$$

Here, N_{cd}, N_{gt}, N_{id}, and N_{ad} are the number of correctly detected berries, manually labeled berries, falsely detected berries, and all detected berries, respectively. In other words, *CD* is the percentage of correctly detected grape berries over the manually labeled grape berries, and *MC* is the percentage of falsely detected berries over all detected berries. The IoU threshold is used to determine whether the detected object is correctly or falsely detected. In the experiment, the threshold is set to 0.5, which follows the approaches from the Pascal VOC Challenges [46]. The annotation application used in this study is the COCO Annotator application [47].

B. EVALUATION OF DATA AUGMENTATION TECHNIQUE

As introduced in Section III.A, the proposed data augmentation technique synthesizes images by removing berries with a circularity below a given threshold. Table 2 shows the number of synthetic images with different circularity thresholds. In terms of the limited diversity of background images and storage resources, we decided to use the circularity threshold 0.6, which can synthesize 956 images from 60 annotated images, for the experiment.

We compare the results with/without using the proposed augmentation method, and six-fold cross-validation was applied. As shown in Table 3, the 60 annotated images are divided into six folds, each of which contains 10 images. The number of images synthesized with the proposed techniques from each fold is also shown in the table.

During cross-validation, 50 original images of five folds and their corresponding synthesized images are used for training, and the 10 original images are used for validation.

Table 4 shows the results of the validation, which is the average of the validation results of all six folds. The HTC [33] model using HRNet [48] as the backbone model was used. We can observe that using the proposed augmentation affords a notable performance over not using augmentation (*MC* decrease of 51.38%). Although the *CD* decreases by 2.17% compared to not using augmentation, the decrease in *MC* is a huge improvement. Fig. 13 shows the detected results without augmentation (left) and with the proposed augmentation (right). The red mask is the detected mask that did not overlap with the ground truth mask (false positive), blue is the detected mask that did overlap with the ground truth mask (true positive), and the green mask is the ground truth that did not overlap with the detected mask (false negative). It is obvious that the proposed method can reduce a large number of false-positive results by trading a small loss of true-positive results. The reason the proposed method can generate images for training an effective model is that it does not destroy the context information in the image. The synthesized image simulates the real pictures taken during the thinning process. The proposed method provides a simple yet efficient approach to prevent model overfitting.

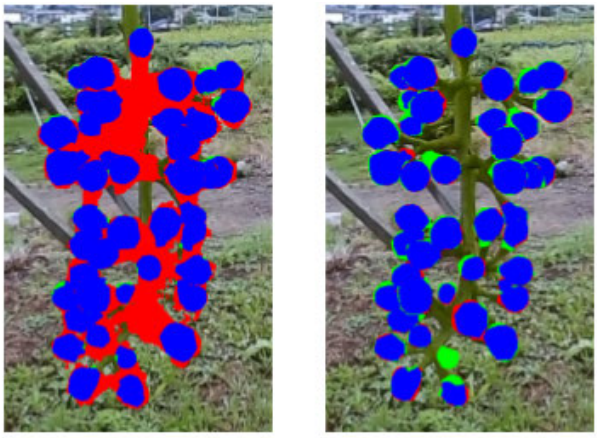

FIGURE 13. Comparison of the detected result between HTC without augmentation (left) and HTC with the proposed augmentation (right). The red mask is the detected mask that did not overlap with the ground truth mask, blue is the detected mask that did overlap with the ground truth mask, and the green mask is the ground truth mask that did not overlap with the detected mask.

TABLE 2. The number of synthesized images and berries with different circularity thresholds.

Circularity threshold	Number of synthesized images	Number of berries
0.5	164	7,325
0.6	**956**	**41,704**
0.7	19,952	970,780

TABLE 3. The number of synthesized images from each fold.

Fold	Number of original images	Number of synthesized images
1	10	185
2	10	130
3	10	103
4	10	114
5	10	332
6	10	92

TABLE 4. Comparison of training the HTC [33] model using HRNet [48] as the backbone model, with and without augmentation.

Methods	*CD* (%)	*MC* (%)
Without augmentation	**98.72**	54.17
With augmentation	96.55	**2.79**

C. EVALUATION OF LOCATION-SENSITIVE HTC MODEL

This section presents the results of the proposed models. Fig. 2 shows the experiment results of the HTC [33] and the proposed location-sensitive models using HRNet [48] as the backbone model. The results show that combining the explicit location feature with the fully connected feature (Fig. 8) improves model accuracy. Integrating the explicit location information with the fully connected features at the new supplementary classification branch and training the model using new supplementary classification loss (Fig. 9) improves model accuracy more than simply integrating the location features in the original branch of the HTC (Fig. 8).

TABLE 5. Comparison of the average processing times of 60 images for different stages shown in Fig. 3 between the HTC [33] model and the proposed models, using HRNet [48] as the backbone model.

Stages	Processing time		
	HTC [33] (s)	Proposed FC (Fig. 8) (s)	Proposed SCLASS (Fig. 9) (s)
Detect working bunch	0.493	0.501	0.517
Extract features	0.363	0.368	0.371
Predict berry number in 3D bunch	0.00847	0.00867	0.01025

TABLE 6. Comparison of the number of trainable parameters and the number of floating-point operations per second (FLOPs) between the HTC [33] model and the proposed models, using HRNet [48] as the backbone model. The size of the input image is 1,280 × 800 pixels.

Model	Parameters (M)	FLOPs (G)
HTC [33]	82.68	516.14
Proposed FC (Fig. 8)	82.68	516.14
Proposed SCLASS (Fig. 9)	83.13	516.58

Furthermore, the 956 synthesized images used for the six-fold cross-validation, as shown in Table 2 and Table 3, are used to evaluate the average number of berries detected from the non-working bunch (Avg_{NWB}) using the metric given in (14).

$$Avg_{NWB} = \frac{1}{Fold}\sum_{i=1}^{Fold} Berries_{NWB}(i) \tag{14}$$

Here, *Fold* is the number of folds, which is 6 in this study, and $Berries_{NWB}(i)$ is the number of non-working bunch berries for each fold i. We compare the proposed SCLASS (Fig. 9) with the conventional HTC [33] using (14). The proposed models can reduce the average number of berries detected from the non-working bunch from 5.33 to 0.33, which can prevent the counting of berries that do not belong to the working bunch. The reason the proposed models can reduce the number of unexpectedly detected berries is that the location features help the model partially learn the location of the object. The explicit location information is what the conventional model uses to specify the feature map location from the pooling layer. However, the explicit location has never been used as an input feature for object classification or prediction in the conventional model. The proposed models make use of a feature that is already available without requiring additional data annotation costs. Especially, the experiment results shown in Table 5 and Table 6 also demonstrate that the proposed models do not consume much more time than the original model. Fig. 14 shows the results of the HTC [33] and the proposed methods. The red circle is the berry that the proposed methods could detect but that the HTC [33] could not.

There exists a trade-off between the accuracy and the computational complexity when selecting a DNN model. We have

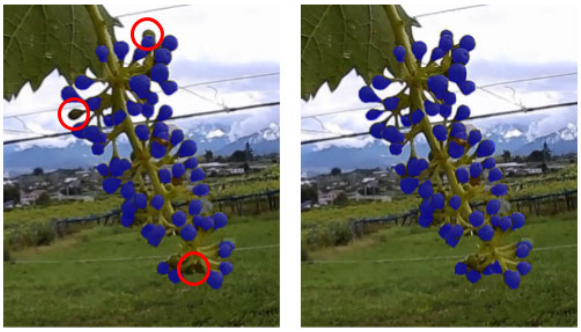

FIGURE 14. Comparison of the detected berry between HTC [33] and the proposed method. The blue mask is the detected berry mask; the red circle is the berry that the proposed method can detect, but that HTC [33] cannot detect.

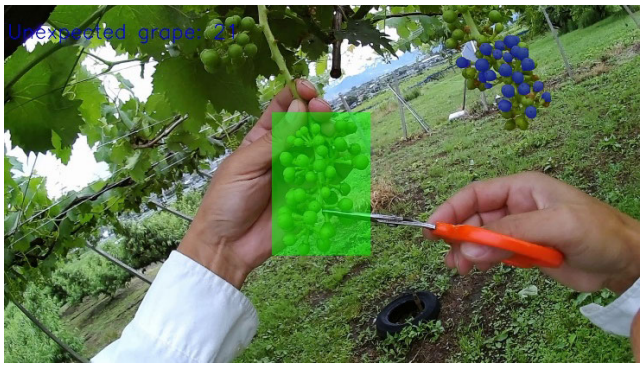

FIGURE 15. The berries (blue) that have been discarded because they do not belong to the working bunch (green).

made an extension for the state-of-the-art instance segmentation model to take advantage of obtaining accurate mask information about individual berries to compute the features required for predicting berry numbers. As shown in Table 5, it takes about 0.9 seconds on average to process one frame on a high-end graphics processing unit (Titan RTX GPU), which makes the method more suitable to be implemented as a remote application. However, during the experiment, we found that the farmers did not actually need to confirm the number of berries in every frame. Therefore, it is possible to implement a user-friendly application even on an embedded AI computing device or mobile device by only computing and visualizing the berry numbers whenever any berries are removed.

D. EVALUATION OF POST-PROCESSING TECHNIQUE

This section shows the evaluation results of post-processing to exclude the berries that do not belong to the working bunch. Fig. 15 shows an example of the berries (in blue color) detected by the location-sensitive HTC model but that were identified as not belonging to the working bunch during post-processing. Fig. 16 shows that the grape berries not belonging to the working bunch are discarded and only the berries in the working bunch are counted. The 2,535 different berry

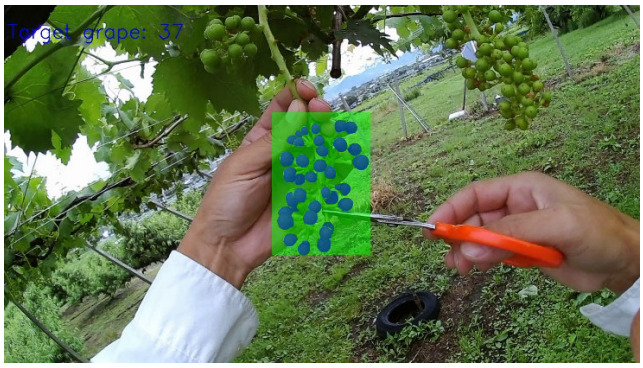

FIGURE 16. The final result (blue) after post-processing, including only the berries in the working bunch.

images are used to evaluate the efficiency of the proposed post-processing method using the metric given in (15).

$$Ab(B_{pd}) = \begin{cases} 1 & : B_{pd} = B_{gt} \text{ and } COUNT(B_{pd}) = 1 \\ 0 & : otherwise \end{cases} \quad (15)$$

Here, Ab is the number of accurately detected working bunch, B_{pd} is the predicted working bunch, and B_{gt} is the ground truth working bunch. We manually check the result for each image and compute the proposed method's average accuracy using (16).

$$A(x)_{avg} = \left(\frac{\sum_{i=1}^{N} x_i}{N} \right) \times 100 \quad (16)$$

Here, $A(x)_{avg}$ is the average accuracy of x, x_i is the accuracy (Ab) for image i, and N is the number of images in this experiment. We found the proposed method could select the working bunch with an accuracy of 100% for all images.

Our experimental results show that combining the proposed location-sensitive models with the post-processing method can well meet the main purpose of our research, that is, the end-to-end automatic counting of berries in a working bunch without counting the berries from other bunches. The proposed method succeeds in tackling this problem.

E. EVALUATION OF AUTOMATIC BERRY NUMBER PREDICTION

To evaluate the regression models using the proposed features described in Section III.C, the dataset was collected by taking images while farmers thin grape berries from start to end. The farmers were asked to rotate the bunch during the process to capture as many images from different perspectives as possible. The actual numbers of berries in the 3D bunch (3D counting) were manually counted as the ground truth. Input features for regression models were extracted using the method proposed in Section III.B. We took a total of 16,607 images, among which 13,285 images were used as training data and the remaining 3,322 images as test data.

The evaluation metric in this experiment is the mean absolute error (MAE) [43], shown below.

$$MAE(X, h) = \frac{1}{m} \sum_{i=1}^{m} \left| h(x^{(i)}) - y^{(i)} \right| \quad (17)$$

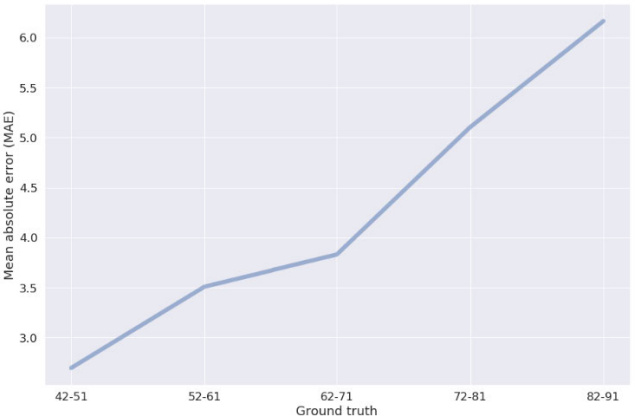

FIGURE 17. MAE as a function of ground truth berry number.

TABLE 7. MAE of berry number prediction for different regression models using the proposed features computed from 2D images.

Regression model	MAE
Kernel ridge regression (KRR) [38]	6.12
Support vector regression (SVR) [39]	4.64
Random forest regression (RFR) [40]	**3.79**
Gradient boosting (GB) [41]	4.57
Stochastic gradient descent (SGD) [42]	5.48
Artificial neural network (ANN) [43]	5.03

where m is the number of instances in the dataset, $x^{(i)}$ is is a vector of all the feature values (excluding the label) of the i^{th} instance in the dataset, $y^{(i)}$ is its label (the desired output value for that instance), X is a matrix containing all the feature values (excluding labels) of all instances in the dataset, h is the regression model (also called a hypothesis), and $MAE(X, h)$ is the cost function measured on the set of example X using hypothesis h.

The regression experiment results of automatically predicting the number of grape berries in the bunch (3D counting) from a single 2D image are shown in Table 7. The results show that using RFR [40] can archive the most accurate estimation (MAE of 3.79). The reason RFR obtains the best accuracy can be explained by the fact that a random forest is good at reducing the variance in the forest estimator by combining diverse trees, which complies with the large variance in features computed from 2D images. For the same bunch, the number of berries visible on a 2D image can vary by more than 10 berries for the images captured from different perspectives. Such a fact makes the berry number-prediction task highly difficult.

In a real practical scenario, when farmers are thinning grapes, the number of berries in a bunch begins at a larger number and reaches a smaller number (target number). Therefore, in the experiment, we computed the MAE as the function of 3D counting to validate the effect of a prediction model during the thinning process. The result is shown in Fig. 17. During the real thinning process, when the number of berries in the bunch is much larger than the target number, the estimation accuracy is relatively unimportant. However, when the number of berries in the bunch approaches the target number, the estimation accuracy becomes critical for avoiding over-thinning. In Fig. 17, MAE decreases when 3D counting decreases. MAE starts from 6.17 for the 3D counting range of 82–91 and decreases to 2.70 for the 3D counting range of 42–51. Because the target number of berries in a bunch for major table grape varieties is less than 40, as shown in Table 1, this experiment result demonstrates that the proposed method can fit real practical scenario usage well. The farmers involved in the experiment are highly satisfied with the performance of the proposed technique.

V. CONCLUSION

The proposed technology is for building a practical application for the real grapevine farm environment. The novel end-to-end berry number prediction technology enables farmers to perform berry thinning efficiently, as it is a crucial task affecting the market value of table grapes. By integrating the location feature into the state-of-the-art instance segmentation DNN model, we succeeded in focusing the berry detection on a working bunch only. The proposed location-sensitive HTC model can also be used for other object detection problems that require detecting a particular object from an image consisting of multiple objects of similar features. Berry number prediction using the originally designed features can also be applied to the image-based counting of other kinds of fruits or vegetables.

We recommend employing the proposed method in the form of a server-side application because the DNN used in this work is huge. The server-side application can afford various devices, such as a mobile application, smart glasses, or an augmented reality headset (e.g., Microsoft HoloLens™) via the application programming interface (API).

REFERENCES

[1] E. Ivorra, A. J. Sánchez, J. G. Camarasa, M. P. Diago, and J. Tardaguila, "Assessment of grape cluster yield components based on 3D descriptors using stereo vision," *Food Control*, vol. 50, pp. 273–282, Apr. 2015.
[2] G. L. Creasy and L. L. Creasy, *Grapes*, 2nd ed., vol. 27. Wallingford, U.K.: CABI, 2018.
[3] M. Karoglan, M. Osrečak, L. Maslov, and B. Kozina, "Effect of cluster and berry thinning on merlot and cabernet sauvignon wines composition," *Czech J. Food Sci.*, vol. 32, no. 5, pp. 470–476, Oct. 2014.
[4] T. T. Santos, L. L. de Souza, A. A. dos Santos, and S. Avila, "Grape detection, segmentation, and tracking using deep neural networks and three-dimensional association," *Comput. Electron. Agricult.*, vol. 170, Mar. 2020, Art. no. 105247.
[5] J. G. A. Barbedo, "Plant disease identification from individual lesions and spots using deep learning," *Biosyst. Eng.*, vol. 180, pp. 96–107, Apr. 2019.
[6] R. S. Jackson, "Grapevine structure and function," in *Food Science and Technology, Wine Science*, R. S. Jackson, Ed., 2nd ed. New York, NY, USA: Academic, 2000, ch. 3, pp. 45–95. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123790620500044, doi: 10.1016/B978-012379062-0/50004-4.
[7] K. Mitsui, "Smart agri—Basic information about the management work of grapes," Dream Farm Co., Ltd., Kofu, Japan, Tech. Rep., 2019.
[8] M. J. C. S. Reis, R. Morais, E. Peres, C. Pereira, O. Contente, S. Soares, A. Valente, J. Baptista, P. J. S. G. Ferreira, and J. B. Cruz, "Automatic detection of bunches of grapes in natural environment from color images," *J. Appl. Log.*, vol. 10, no. 4, pp. 285–290, Dec. 2012.
[9] S. Nuske, K. Wilshusen, S. Achar, L. Yoder, S. Narasimhan, and S. Singh, "Automated visual yield estimation in vineyards," *J. Field Robot.*, vol. 31, no. 5, pp. 837–860, Sep. 2014.
[10] S. Liu, M. Whitty, and S. Cossell, "A lightweight method for grape berry counting based on automated 3D bunch reconstruction from a single image," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA), Workshop Robot. Agricult.*, May 2015, p. 4.

[11] A. Aquino, B. Millan, D. Gaston, M.-P. Diago, and J. Tardaguila, "vitisFlower: Development and testing of a novel android-smartphone application for assessing the number of grapevine flowers per inflorescence using artificial vision techniques," *Sensors*, vol. 15, no. 9, pp. 21204–21218, Aug. 2015.
[12] L. Luo, Y. Tang, X. Zou, C. Wang, P. Zhang, and W. Feng, "Robust grape cluster detection in a vineyard by combining the AdaBoost framework and multiple color components," *Sensors*, vol. 16, no. 12, p. 2098, Dec. 2016.
[13] Š. Kohek and D. Strnad, "Interactive synthesis of self-organizing tree models on the GPU," *Computing*, vol. 97, no. 2, pp. 145–169, Feb. 2015.
[14] S. Liu, S. Cossell, J. Tang, G. Dunn, and M. Whitty, "A computer vision system for early stage grape yield estimation based on shoot detection," *Comput. Electron. Agricult.*, vol. 137, pp. 88–101, May 2017.
[15] A. Aquino, I. Barrio, M.-P. Diago, B. Millan, and J. Tardaguila, "VitisBerry: An android-smartphone application to early evaluate the number of grapevine berries by means of image analysis," *Comput. Electron. Agricult.*, vol. 148, pp. 19–28, May 2018.
[16] A. Aquino, B. Millan, M.-P. Diago, and J. Tardaguila, "Automated early yield prediction in vineyards from on-the-go image acquisition," *Comput. Electron. Agricult.*, vol. 144, pp. 26–36, Jan. 2018.
[17] R. Pérez-Zavala, M. Torres-Torriti, F. A. Cheein, and G. Troni, "A pattern recognition strategy for visual grape bunch detection in vineyards," *Comput. Electron. Agricult.*, vol. 151, pp. 136–149, Aug. 2018.
[18] A. K. Nellithimaru and G. A. Kantor, "ROLS : Robust object-level SLAM for grape counting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 2648–2656.
[19] L. Zabawa, A. Kicherer, L. Klingbeil, A. Milioto, R. Topfer, H. Kuhlmann, and R. Roscher, "Detection of single grapevine berries in images using fully convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 2571–2579.
[20] R. Rudolph, K. Herzog, R. Töpfer, and V. Steinhage, "Efficient identification, localization and quantification of grapevine inflorescences and flowers in unprepared field images using fully convolutional networks," *Vitis J. Grapevine Res.*, vol. 58, no. 3, pp. 95–104, Aug. 2019.
[21] X. Liu, S. W. Chen, S. Aditya, N. Sivakumar, S. Dcunha, C. Qu, C. J. Taylor, J. Das, and V. Kumar, "Robust fruit counting: Combining deep learning, tracking, and structure from motion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1045–1052.
[22] M. Rahnemoonfar and C. Sheppard, "Deep count: Fruit counting based on deep simulated learning," *Sensors*, vol. 17, no. 4, p. 905, Apr. 2017.
[23] I. Sa, Z. Ge, F. Dayoub, B. Upcroft, T. Perez, and C. McCool, "DeepFruits: A fruit detection system using deep neural networks," *Sensors*, vol. 16, no. 8, p. 1222, Aug. 2016.
[24] S. Amatya and M. Karkee, "Integration of visible branch sections and cherry clusters for detecting cherry tree branches in dense foliage canopies," *Biosyst. Eng.*, vol. 149, pp. 72–81, Sep. 2016.
[25] R. Roscher, K. Herzog, A. Kunkel, A. Kicherer, R. Töpfer, and W. Förstner, "Automated image analysis framework for high-throughput determination of grapevine berry sizes using conditional random fields," *Comput. Electron. Agricult.*, vol. 100, pp. 148–158, Jan. 2014.
[26] S. Nuske, K. Wilshusen, S. Achar, L. Yoder, S. Narasimhan, and S. Singh, "Automated visual yield estimation in vineyards," *J. Field Robot.*, vol. 31, no. 5, pp. 837–860, Sep. 2014.
[27] O. Mirbod, L. Yoder, and S. Nuske, "Automated measurement of berry size in images," *IFAC-PapersOnLine*, vol. 49, no. 16, pp. 79–84, Jan. 2016.
[28] A. Aquino, M. P. Diago, B. Millán, and J. Tardáguila, "A new methodology for estimating the grapevine-berry number per cluster using image analysis," *Biosyst. Eng.*, vol. 156, pp. 80–95, Apr. 2017.
[29] F. Schöler and V. Steinhage, "Automated 3D reconstruction of grape cluster architecture from sensor data for efficient phenotyping," *Comput. Electron. Agricult.*, vol. 114, pp. 163–177, Jun. 2015.
[30] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2999–3007.
[31] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Mar. 2017, pp. 2980–2988.
[32] Z. Cai and N. Vasconcelos, "Cascade R-CNN: High quality object detection and instance segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Nov. 28, 2019, doi: 10.1109/TPAMI.2019.2956516.
[33] K. Chen, W. Ouyang, C. C. Loy, D. Lin, J. Pang, J. Wang, Y. Xiong, X. Li, S. Sun, W. Feng, Z. Liu, and J. Shi, "Hybrid task cascade for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4969–4978.
[34] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Trans. Graph.*, vol. 36, no. 4, p. 107, 2017.
[35] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.
[36] J. J. Friel, *Practical Guide to Image Analysis*. New York, NY, USA: ASM, 2000. [Online]. Available: https://www.amazon.com/Practical-Guide-Image-Analysis-Friel/dp/0871706881
[37] D. A. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.
[38] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. Cambridge, MA, USA: MIT Press, 2012.
[39] C. M. Bishop, *Pattern Recognition and Machine Learning*, vol. 4, no. 4. New York, NY, USA: Springer-Verlag, 2006. [Online]. Available: https://www.springer.com/gp/book/9780387310732#aboutBook
[40] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001.
[41] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Stat.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001. [Online]. Available: https://www.jstor.org/stable/2699986
[42] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. COMPSTAT*, 2010, pp. 177–186.
[43] A. Géron, *Hands-on Machine Learning With Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. Newton, MA, USA: O'Reilly, 2019.
[44] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Rev.*, vol. 60, no. 2, pp. 223–311, Jan. 2018.
[45] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch SGD: Training ImageNet in 1 hour," Jun. 2017, *arXiv:1706.02677*. [Online]. Available: http://arxiv.org/abs/1706.02677
[46] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille, "The role of context for object detection and semantic segmentation in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 891–898.
[47] J. Brooks. *COCO Annotator*. Accessed: 2019. [Online]. Available: https://github.com/jsbroks/coco-annotator
[48] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5686–5696.

PRAWIT BUAYAI received the B.E. and M.E. degrees in computer engineering from Khon Kaen University (KKU), Thailand, in 2013 and 2016, respectively. He is currently pursuing the Ph.D. degree in computer engineering with the University of Yamanashi, Japan. His main research interests include the application of artificial intelligence technology to improve accessibility and productivity in agriculture.

KANDA RUNAPONGSA SAIKAEW received the B.S. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 1997, and the M.S. and Ph.D. degrees in computer science and engineering from the University of Michigan at Ann Arbor, Ann Arbor, MI, USA, in 1999 and 2003, respectively. In 2003, she joined the Department of Computer Engineering, Khon Kaen University, as a Lecturer, and became an Associate Professor, in 2015. Her current research interests include artificial intelligence and smart agriculture.

XIAOYANG MAO (Member, IEEE) received the B.S. degree in computer science from Fudan University and the M.S. and Ph.D. degrees in computer science from The University of Tokyo. She is currently a Professor with the Department of Computer Science and Engineering, University of Yamanashi, Japan, and an Adjunct Professor with the College of Computer Science, Hangzhou Dianzi University, China. Her current research interests include image processing, visual perception, non-photorealistic rendering, and their applications to e-health and smart agriculture. She received the Computer Graphics International Career Achievement Award, in 2018.