Rapid Image Detection of Tree Trunks Using a Convolutional Neural Network and Transfer Learning

Ting-ting YANG, Su-yin ZHOU, Ai-jun XU*

Abstract—**The rapid detection of tree trunks is key to forest automation, inventory, and monitoring, enabling the use of tree-harvesting robots capable of navigation, tree counting, and tree measurement. In this paper, we propose a method called yolov3_trunk_model (Y3TM) to detect trunks rapidly using a convolutional neural network (CNN) and transfer learning. We use an enhanced yolov3 for object detection and an improved prediction strategy using feature pyramid networks (FPNs) for classification and boundary box determination of the tree trunks. Experimental results showed that our Y3TM offers a greatly improved recall rate of over 93% with a drastically average detection time of 0.3 s.**

Keywords—**Convolutional neural network, transfer learning, object detection, trunk, deep learning**

I. INTRODUCTION

In recent years, the development of smart forestry has led to more research into forestry information including intelligent forestry monitoring (the number of standing trees, deforestation rate, etc.), trunk diameter at breast height (DBH), tree height, and forest growing stock. Tree trunks detection is helpful for these studies. However, research into tree image processing is still in the development stage. Chen et al. extract structural parameters of single standing trees by using trunk detection-aided mean shift clustering techniques to conduct a forest information survey [1]. More broadly, recognizing and locating tree trunks is the first critical operation for agroforestry harvesting robots [2]. For example, Kolb et al. use a tree trunk detection system for an autonomous tree-felling robot [3]. Such robotic devices must be capable of identifying the position, size, and orientation of trees in a forest [4]. Identifying and locating the tree trunk [1] are also prerequisites for automated fertilization, trunk injection, pesticide application, and touchless tree measurement [5-6].

The majority of current research methods employed in trunk detection are Laser Imaging Detection And Ranging (LiDAR), the color space model, color space, and classifiers. Bargoti et al. initially used point cloud LiDAR data to obtain a rough estimation of trunk candidates [2]. These candidates were then projected into the camera images, where pixel-wise classification was used to update their likelihood of being a trunk. A hidden semi-Markov model used contextual information from an orchard to perform trunk detection. This method is suitable for single trunk identification but cannot recognize multiple trunks.

Trunk detection methods commonly use visual system information to distinguish significant areas from complex backgrounds. Guan et al. realized trunk detection by constructing a visual saliency map in the Lab (CIELAB) color space and using the hue component of HSV (hue, saturation, value) to enhance color contrast [5]. However, the differences in trunk color and texture across tree species and the similar color between a trunk and the background make detection effect using the color space difficult. To solve this problem, Chen et al. combined color histograms with training classifiers to improve trunk detection performance [1]. Chen's method first used oriented gradient and support vector machine histograms to train the initial trunk classifier. It then extracted the grayscale histogram features of trunk and non-trunk images to optimize the classifier. Finally, it employed the Roberts Edge Detector to extract the trunk's gradient histogram features to improve identification accuracy. Juman et al. combined the color space model with depth information for trunk detection [7]. Pre-processing used color space combination and segmentation to eliminate the ground not covered by trees from the images and then fed the resulting image into a cascade detector to identify the locations of trunks in the image. Depth information was obtained via a Microsoft KINECT sensor to further increase the accuracy of the detector.

Differences make batch identification and location of trunks difficult, such as tree species, environment, and the number

Manuscript received December 24, 2019; revised September 18, 2020. This work was supported by the National Natural Science Foundation of China[31670641] and Zhejiang Science and Technology Key R&D Program Funded Project [2018C02013].

#This author contributed equally to this work and should be considered the co-first author.

The co-first author:

Ting-tingYANG is with the Zhejiang Agriculture and Forestry University, Hangzhou, 311300, Zhejiang, China(e-mail:917251944@qq.com).

Su-yin ZHOU is with the Zhejiang Agriculture and Forestry University, Hangzhou, 311300, Zhejiang (e-mail:zsy197733@163.com).

*Corresponding Author: Ai-jun XU

Ai-jun XU is with the Zhejiang Agriculture and Forestry University, Hangzhou, 311300, Zhejiang, China (e-mail: xuaj1976@163.com).

and location of trees in each image. Therefore, we must establish a method for quickly detecting trunks to reduce manual labor and improve the efficiency of fertilization, object picking, and non-contact DBH measurement by robots. Similar to the introduction of deep learning in fields such as medicine [8], we apply deep learning to improve forest management and analysis. In recent years, convolutional neural networks(CNNs) have made great strides in image classification and recognition [9-11]. The application of CNNs to object detection falls into two categories: two-stage detection and single-stage detection. Two-stage object detection combines a regional proposal with CNN classification using R-CNN, SPP-NET, or Faster R-CNN [12-14]. The detection accuracy is gradually promoted in this method. In contrast, a single-stage object detection network converts object detection into a regression problem and utilizes detection methods such as yolov1-yolov3 and Single Shot MultiBox Detector (SSD) [15-17]. The result is significantly faster detection.

Researching articles on trunk detection inspection in recent years, found that these methods are mainly adopted traditional ways, and the efficiency and accuracy still need to be improved. Target object detection based on CNNs can help solve these problems. In natural environments, both dense forests and sparse urban street trees, we propose a new model for rapid tree trunk detection for intelligent monitoring of forestry resources, forestry harvesting robots, and computer vision-based tree measurement using a CNN and transfer learning. We proposed our trunk detection model on yolov3. We transfer the object detection knowledge acquired from yolov3 on the Pascal VOC dataset to our Y3TM (yolov3_trunk_model) through transfer learning. We designed and optimized Y3TMs with different depth convolutions and input sizes. The tree detection results can be used for rapid counting in forest resource survey, and also have reference value for navigation of forestry robot.

II. Experimental dataset

A. *Original Trunk Dataset*

We used image data from photographs taken in the natural environment of Hangzhou, Zhejiang province, on the southeast coast of China (119.72E, 30.23N), from October to December 2018. The tree species included poplar, sequoia, Chinese parasol, willow, and ginkgo. We used a mobile phone camera with a resolution of 2448 × 3264 pixels to collect a total of 812 original RGB images for the trunk dataset. Multiple factors affected the training of the trunk dataset. We considered the following factors: (1) Illumination conditions: we took photographs under a variety of lighting conditions, including direct sunlight, back-lighting, and shade. (2) Tree species and age: we selected trunks for different species and DBHs (0.1–0.7 m). (3) Distance: we varied the shooting angle and distance between 3 and 10 m.. (4) Occlusions: the trunks in our dataset had different degrees of occlusion. (5) Other objects: the collected dataset contained vertical objects similar to trunks, such as streetlights and telephone poles, with each image containing between 1 and 15 objects. In the forest, the two common types of objects are similar to the vertical trunk. Other, non-similar objects were not considered.

B. *Trunk Dataset Augmentation*

The original trunk dataset of 812 images collected in the wild was insufficient to support the deep convolution network in fully learning the object characteristics. To reduce over- and under-fitting in the training stage and to improve the detection performance, we applied operations such as mirror-transformation, cropping, and rotation to the original images. Thus, we augmented the trunk dataset to produce a final dataset of 1198 images [18]. Trunks accounted for 90% of the total images, with streetlights and telephone poles making up the remaining 10%. We randomly selected 80% of the images for training, with the remaining 20% used for testing (see Fig. 1(a)). We have operated these images offline and then fed all the images to the learning algorithm.

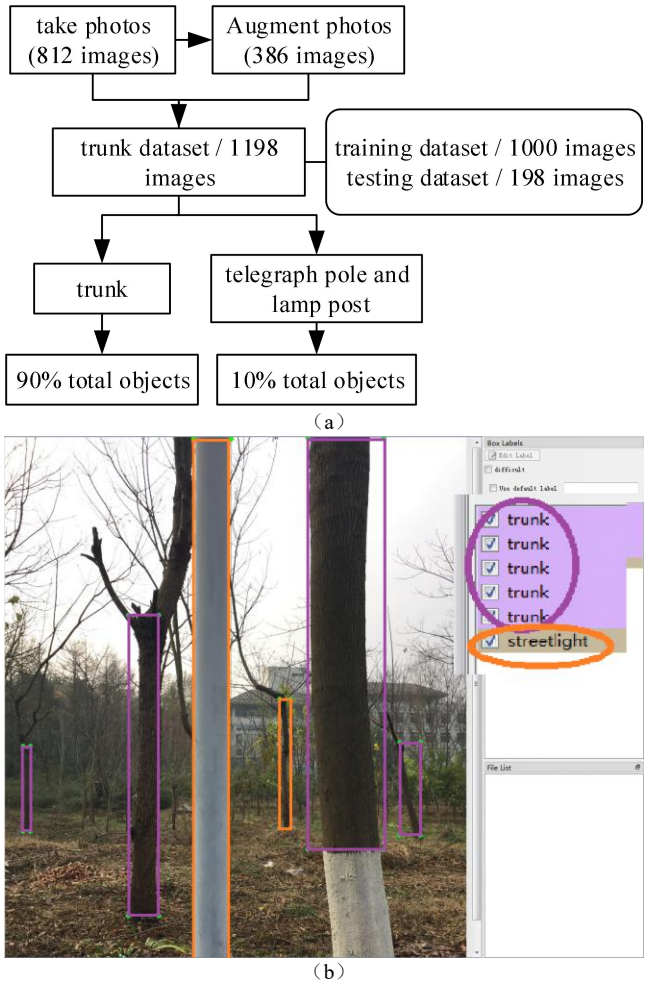


Fig. 1. (a) Data distribution and (b) labeled image as ground truth.

C. *Dataset Labeling*

Even those objects that are far away from the camera, we manually marked all the trunks in the images to increase the probability of detecting every trunk in each image. Since the trunks have no standard boundaries, the markings were made from the first branch down to the visible above-ground roots (the white ash parts of the trunk were not marked to reduce the influence of feature learning). The trunk dataset images were marked according to the Pascal VOC2012 dataset requirements by using the labeling tool and then divided into the training and testing datasets. We labeled each of these detection objects as the trunk, telephone pole, or streetlight (see Fig. 1.(b)).

III. METHOD

A. Transfer Learning

Some deep convolutional neural network models have been fully trained on large datasets to learn the large number of image features required for target detection, such as ResNet, VGG, and Darknet. Using the transfer learning idea [19], we applied feature knowledge to Y3TM. This improvement not only shortened the training time but also accelerated the convergence of the network. There are two primary transfer learning methods. The first one adopts the structure of the pre-trained model. All the weights are first randomized and then trained according to datasets. The other is parameter transfer, training a specific layer, and freezing other layers. The weights of some layers at the beginning of the model are kept unchanged, and then the subsequent layers are retrained to obtain new weights. We selected the second transfer learning method with fine-tuning parameters.

B. Model Structure

We have optimized the Y3TM to quickly identify whether there are trunks in the image and then to locate them. Therefore, from the perspective of increasing the recall rate and decreasing detection time, we obtain a new convolutional neural network by improving yolov3. The specific network structure is shown in Fig. 2. And in Yolo v3, it further adopted three feature graphs of different scales to detect objects. Due to the multiple down-sampling, the sensitivity field of the feature map was relatively large, so it was suitable for detecting objects with large-scale in the image. After several convolutional layers, the feature map was down-sampled 16 times than the input image. It had a medium-scale receptive field and was suitable for detecting the medium-scale objects. After up-sampling, a feature map that was down-sampling 8 times relative to the input image was finally obtained. It had the smallest receptive field and was suitable for detecting small-scale objects.

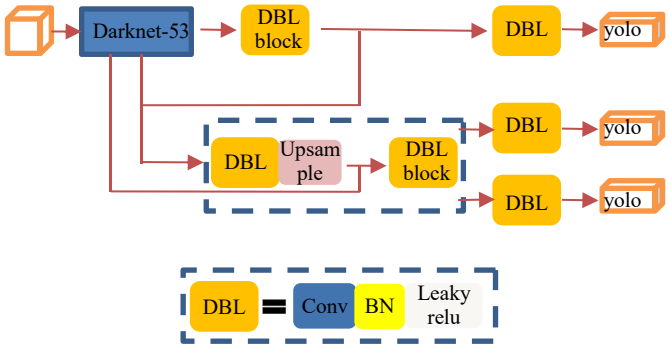


Fig. 2. Y3TM structure.
Note: RGB images of all sizes are acquired and set to Sa × Sb pixels in the network structure. The Darknet-53 network is composed of a convolution layer DBL and residual layer res. All convolutional strides are defaulted to (1, 1), and padding is the same or valid, while strides are (2, 2). DBL block = DBL * m, where m is an integer (the DBL of the latter DBL of DBL_block does not contain the BN and leaky ReLU layers). The figure is re-drawn based on https://blog.csdn.net/leviopku/article/details/82660381 .

Y3TM uses the basic Darknet-53 algorithm in the feature extraction layer and introduces a residual structure to prevent gradient dispersion. The object detection strategy of the feature interaction layer adopts the multiple-scale fusion method to express the fine-grained features of a small object at a greater distance from the camera lens in an image. Y3TM performs tensor size transformation by increasing the step size of the convolution kernel, eliminating pooling and fully connected layers, and extracting image features by relying on a large number of 1×1 and 3×3 convolution kernels. The main differences between Y3TM and yolov3 are as follows.

(1) Modified input size. The high-resolution features of the image obtained from downsampling the input size are more refined and beneficial to the feature expression of the trunk dataset. Considering the processing speed of the computer and the generation of odd × odd grids, 480 × 480 pixels is set as the input size (the source network input size is 416 × 416 pixels).

(2) DBL_block size. We selected the DBL_block according to our requirements (e.g., efficiency and higher recall rate). It is located in the feature interaction layer for predicting the boundary box (see Fig. 2). Since there is no standard boundary box for trunks, and we aim to identify all trunks in an image, the DBL_block depth can be decreased or increased as required. The DBL_block shown in Fig. 3 is divided into three independent unequal convolution groups. Each group adopts different convolution kernels to change the depth of the network. Using 1×1 convolution kernel compression parameters and a 3×3 convolution kernel to increase the channel's filters results in enhanced semantic feature extraction.

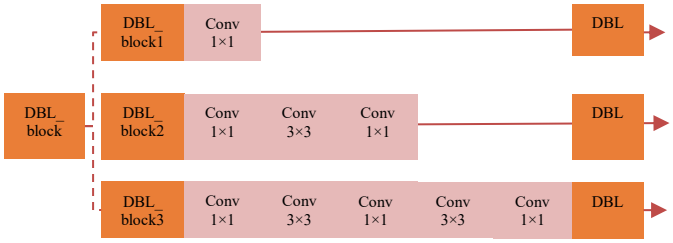


Fig. 3. Different DBL_block.

(3)Network was fine-tuning. To speed Y3TM's convergence on the trunk dataset and obtain a better training effect than yolov3, the model parameters of the source network need to be fine-tuned. We modified the trunk dataset parameter *cls_num* to 3, reflecting the 3 labels for the trunk, streetlight, and telephone pole. We changed the number of filters of the three yolo detection layers to 24 and reset all of the super-parameters of the model.

(4)The Y3TM includes Y3TM_A, Y3TM_B and Y3TM_C. Their differences that input size and DBL_block (TABLE I).

TABLE I
Difference detection models

Model	DBL_block	Input size
Y3TM_A	DBL_block1	416×416 480×480
Y3TM_B	DBL_block2	416×416 480×480
Y3TM_C yolov3	DBL_block3	480×480 416×416

C. *Object Bounding Box and Category Prediction*

Y3TM object detection [20] predicts location by using the shift of center point b (b_x, b_y) of the prediction box on the feature map relative to the upper-left coordinate (c_x, c_y) of the grid, using the logistic function to constrain $\sigma(t_x)$ and $\sigma(t_y)$ to within (0, 1). We first obtain nine prior anchor boxes by k-means clustering (p_w and p_h are the width and height, respectively, of the preset anchor boxes mapped to the feature map) and then divide them into three scales corresponding to each feature layer. We calculate the width and height (b_w, b_h) of the prediction box indirectly using the scaling factors t_w and t_h. We normalize the anchor box size relative to the input image size (S_a, S_b) and then calculate the normalized coordinates of b (b_x, b_y, b_w, b_h) as

$$b_x = \frac{(\sigma(t_x) + c_x)}{S_a}, b_y = \frac{(\sigma(t_y) + c_y)}{S_b}. \quad (1)$$

$$b_w = \frac{p_w e^{t_w}}{S_a}, b_h = \frac{p_h e^{t_h}}{S_b}. \quad (2)$$

The confidence σ(to) reflects the accuracy of the object location:

$$\sigma(t_o) = P(obj) \times IOU\ (b, obj). \quad (3)$$

where $P(obj) \in \{0,1\}$ and IOU(b, obj) represent the probability of detecting objects and the degree of coincidence between the prediction box and the real box, respectively.

The location and confidence of the object t_i are calculated as $t_i = (t_x, t_y, t_w, t_h, t_o)$. In the logistic regression, the priori boxes are scored on the inside of the anchor boxes, with the best selected for prediction. This reduces the amount of calculation required.

D. *Tree Trunk Dataset Training*

Y3TM optimizes parameters using a multi-objective loss function. When t_{ik}^j=1,it indicates that the center point of the real box g^j of the jth target on one trunk image falls into the ith grid and matches the anchor box of the kth predictor. Thus, the k value of the ith grid is the predictor of the object j [21]. One of the goals of model optimization is to make the prediction box of predictor k closer to g^j. The loss function L_1 of the object is expressed as

$$\begin{aligned} L_1 &= \sum_j^M t_{ik}^j \sigma(t_x^{j^*}) \log(\sigma(t_x^{ik})) + (1 - \sigma(t_x^{j^*})) \log(1 - \sigma(t_x^{ik})) \\ &+ (\sum_j^M t_{ik}^j \sigma(t_y^{j^*}) \log(\sigma(t_y^{ik})) + (1 - \sigma(t_y^{j^*})) \log(1 - \sigma(t_y^{ik}))) \\ &+ \sum_j^M (\sigma(t_w^{ik}) - \sigma(t_w^{j^*}))^2 + \sum_j^M (\sigma(t_h^{ik}) - \sigma(t_h^{j^*}))^2 \end{aligned} \quad (4)$$

In Eq. (4), M is the actual number of objects in the current image; t_x^{ik}, t_y^{ik}, t_w^{ik}, and t_h^{ik} represent the predicted values of the bounding box of the kth predictor of grid i; $\sigma(t_w^{j^*})$ and $\sigma(t_h^{j^*})$ represent the normalized offset of the center point of object j relative to the coordinates of the upper-left corner of the grid i; and $\sigma(t_w^{j^*})$ and $\sigma(t_h^{j^*})$ are the nonlinear scaling factors of the anchor box of the kth predictor for g^j.

Another model goal is to optimize the predictor's to make the co-nfidence close to 1. The loss function L_2 for this goal is

$$L_2 = \sum_j^M t_{ik}^j \sigma(t_o^{j^*}) \log(\sigma(t_o^{ik})) + (1 - \sigma(t_o^{j^*})) \log(1 - \sigma(t_o^{ik})) \quad (5)$$

where t_o^{ik}represents the confidence of the prediction value of the kth predictor of the ith grid.

The loss function L_3 corresponding to the value of the category predictor is

$$\begin{aligned} L_3 &= \sum_j^M t_{ik}^j (P(c_n^{j^*} \mid obj)) \log(P(c_n^{ik} \mid obj)) \\ &+ (1 - P(c_n^{j^*} \mid obj)) \log(1 - P(c_n^{ik} \mid obj)) \end{aligned} \quad (6)$$

where $P(c_n^{ik}|obj)$ is the prediction value probability, and $P(c_n^{j^*}|\text{obj})$ is the actual category probability.

Finally, the multi-target loss function L for Y3TM becomes

$$L(\{t_i\}) = L_1 + L_2 + L_3 \quad (7)$$

where $\{t_i\}$ represents the set of all prediction values of the grid.

IV. Results and Discussions

A. Experimental Environment

We performed model training and testing using the TensorFlow framework running on a system with an Intel Xeon E-2134 CPU at 3.50 GHz, 16 GB RAM, and a graphics with an Nvidia Quadro P1000 GPU and 4 GB of RAM. The software environment consisted of CUDA Toolkit 9.0, CUDNN v7.0; Python 3.5.2, TensorFlow-GPU 1.9.0; and the Ubuntu 16.04 operating system. Both training and testing models use GPU acceleration.

B. Results of Model Training

We used the gradient descent method with momentum [22] to train the trunk detection network. Considering the hardware performance and training time, we set the model super-parameters as follows: batch size 32, momentum 0.9, weight decay 0.0005, and learning rate 0.0001. After the 30th epoch, the loss of each model stabilized. To evaluate detection performance, we used the error rate, IOU, and recall rate (IOU ≥ 0.3) as indicators [23].

TABLE II
The results obtained when training Y3TM and yolov3.
(A)Error rate

Error rate(%)					
Model	Input size	Multi-scale prediction (%)			Average (%)
		Large	Medium	Small	
Y3TM_A	416×416	3.48 (—)	4.76 (—)	5.91 (—)	4.42
	480×480	3.27 (—)	4.64 (—)	5.52 (—)	4.27
Y3TM_B	416×416	3.24 (—)	4.84 (—)	5.71 (—)	4.39
	480×480	3.09 (—)	4.72 (—)	5.24 (—)	4.25
Y3TM_C	480×480	3.38 (—)	5.39 (—)	7.62 (—)	4.87
yolov3	416×416	4.09 (—)	6.15 (—)	6.97 (—)	5.53

(B)Recall rate

Recall rate (IOU ≥ 0.3) (%)					
Model	Input size	Multi-scale prediction (%)			Average (%)
		Large	Medium	Small	
Y3TM_A	416×416	87.78 (30.98)	96.89 (63.94)	82.31 (5.08)	93.32
	480×480	88.24 (30.47)	97.93 (64.40)	81.67 (5.13)	94.14
Y3TM_B	416×416	87.93 (30.79)	97.76 (64.38)	83.89 (4.83)	94.06
	480×480	88.19 (30.34)	98.74 (64.83)	83.98 (4.83)	94.83
Y3TM_C	480×480	88.20 (31.08)	98.50 (64.16)	85.13 (4.76)	94.66
yolov3	416×416	85.08 (32.16)	96.98 (62.94)	80.80 (4.90)	92.36

(C)IOU

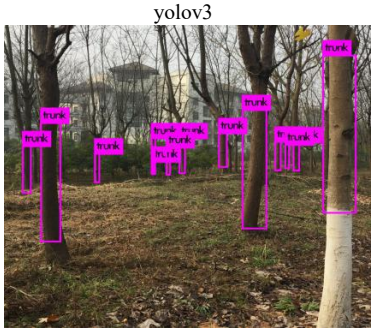
Model	Input size	IOU (%) Multi-scale prediction (%) Large	Medium	Small	Average (%)
Y3TM_A	416×416	62.98 (—)	70.81 (—)	61.93 (—)	67.93
	480×480	63.17 (—)	70.94 (—)	61.71 (—)	68.11
Y3TM_B	416×416	63.20 (—)	70.97 (—)	62.56 (—)	68.17
	480×480	94.83	63.34 (—)	71.91 (—)	62.80 (—)
Y3TM_C	480×480	94.66	63.15 (—)	70.58 (—)	62.91 (—)
yolov3	416×416	92.36	62.82 (—)	69.40 (—)	61.84 (—)

Note: We say that the unsampled prediction is a small-scale prediction, the first upsampling prediction is a medium-scale prediction, and the second upsampling prediction is a large-scale prediction. The values in parentheses represent the probability that the large-scale, medium-scale, and small-scale objects are detected during the training process, and (—) represents that the corresponding values are equal.

These four trunk detection models have high average recall and low average error rates. The values in parentheses show that the average probability of objects predicted at the medium-scale prediction was about 64%, which greatly affects the average recall rate. The average error rate of large-scale prediction was better than that of the small- and medium-scale prediction because of advanced features. The comprehensive performance of the small-scale prediction with no obvious feature was poor and had a probability of only about 5%.

In comparing the models with identical network structures and input sizes in TABLE II, we found that the performance indicators of models with high resolution were improved by different degrees. For example, compared with yolov3, Y3TM achieved the highest average recall rate and IOU that was better by 2.47% and 1.96%, respectively, indicating that the Y3TM better identified small objects at a long distance (see Fig. 4). Moreover, the average IOU of Y3TM_B was 1.96% higher than that of yolov3. As shown in Fig. 4, the predicted boundary box deviations of Y3TM_B were smaller, but close and small objects are missed.

yolov3



Y3TM_C

Fig. 4. Training results with different input sizes.

In the detection model with the same input size but different DBL_block value, Y3TM_B had an additional 1×1 and 3×3 convolution kernel on each prediction scale compared to Y3TM_A, deepening the network learning to obtain more advanced features. TABLE II shows the relative performance improvement of Y3TM_B over Y3TM_A. For the same input size, yolov3 had some missing objects and had some deviations in the fitting degree of the object prediction boundary boxes (see Fig. 5). Y3TM_B obtained strong features and, compared to Y3TM_A, showed slightly better detection ability for occluded objects and small objects at a distance.

yolov3 Y3TM_A (480×480) Y3TM_B (480×480)

Fig. 5. Training results with different input sizes.

To analyze the trunk detection ability of Y3TM during the training process, we calculated the average recall rate for different IOU thresholds. As shown in Fig. 6 (input size 480 × 480 pixels), when $0.3 \leq IOU \leq 0.6$, the average recall rate of Y3TM and yolov3 was high and gradually declined, indicating that the model performance was adequate and stable in this interval. When $0.6 < IOU \leq 1$, the detection ability of the model decreased rapidly. The boundary boxes were uncertain and resulted in small predicted IOU values. Fig. 6 shows that the average recall rate of Y3TM_C was higher than that of the Y3TM_A and Y3TM_B models, which had equal detection performance.

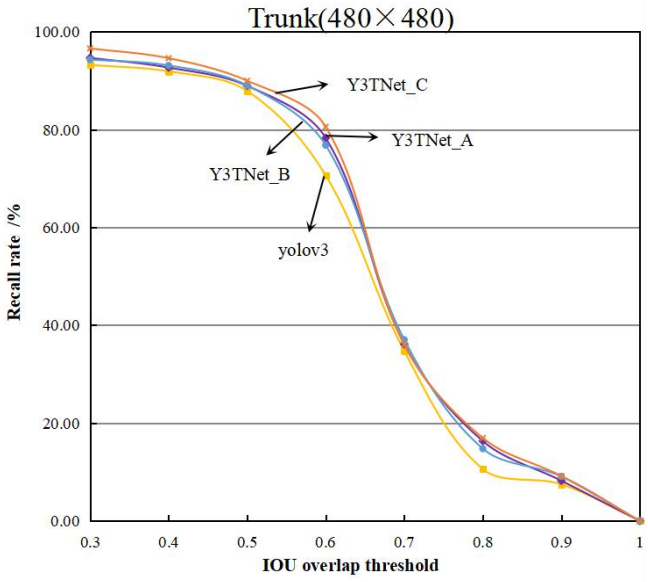


Fig. 6. Average recall rate with different IOU thresholds.

C. Model Testing and Comparison

Using the same configuration as in the training environment, we tested the remaining 20% of the trunk dataset using different detection models. When multiple predictors of the network predict the objects, the non-maximum suppression merging

method is used to predict results [24]. TABLE III shows the testing results of the different detection models with the same super-parameters.

TABLE III
Comparative analysis of the detection ability of different models

Model	Size (pixels/MB)	Average error rate (%)	Average recall rate (IOU≥0.3) (%)	Time (s)
Y3TM_A	416×416/191	5.67	90.21	0.30
	480×480/204	5.04	91.17	0.41
Y3TM_B	416×416/276	5.43	92.65	0.63
	480×480/301	4.98	93.13	0.68
Y3TM_C	480×480/354	5.36	93.60	0.87
yolov3	416×416/322	6.49	89.18	0.72

Tables II and III show that the comprehensive testing and training performance of Y3TM are similar and stable, indicating that our model had better generalization ability than yolov3. Y3TM_A uses a simpler algorithm and had the fastest detection time (0.30 s). The average error rate of Y3TM_B was lowest among the models (4.98%) and was 1.51% lower than that of yolov3. Furthermore, the average recall rate was significantly higher than yolov3's. The average recall rate of Y3TM_C was 4.42% higher than that of yolov3, and the average error rate was 1.13% lower. The testing results in Fig. 7 and TABLE III show that yolov3 had the highest average error rate (there is the problem that the telephone pole is mistakenly classified as a trunk), with larger object boundary box deviations and the lowest average recall rate among all models (the trunks at a distance are missed). As shown in Fig. 7, Y3TM_C had the smallest object boundary box deviations and better detection of small targets at a distance than other models.

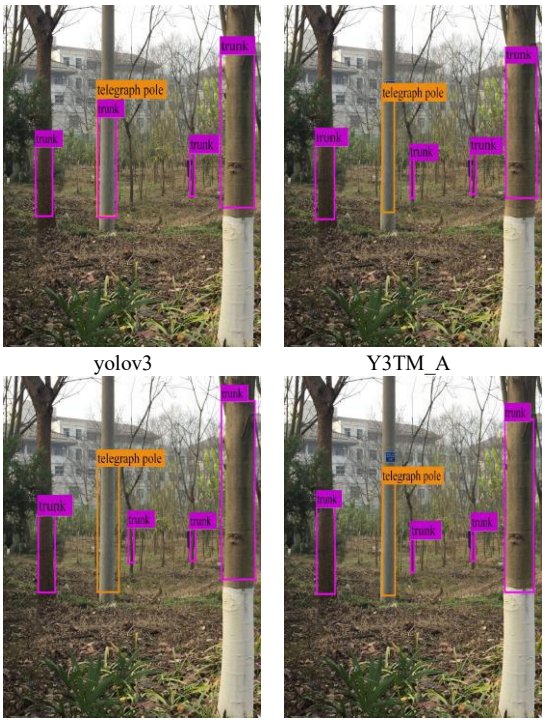
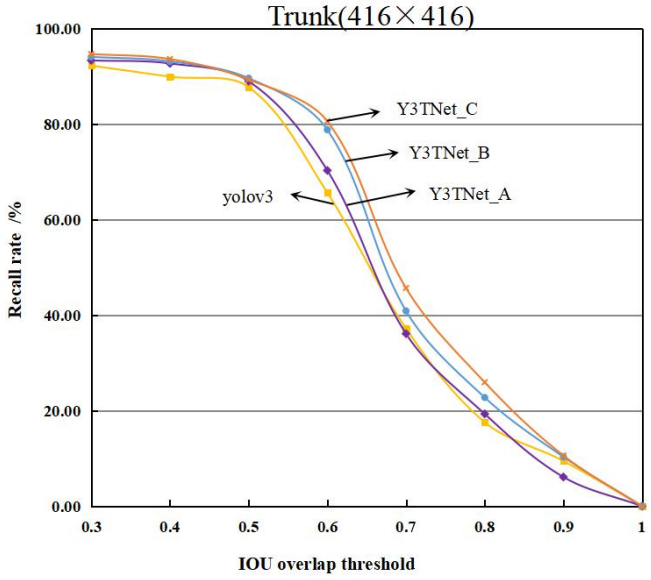
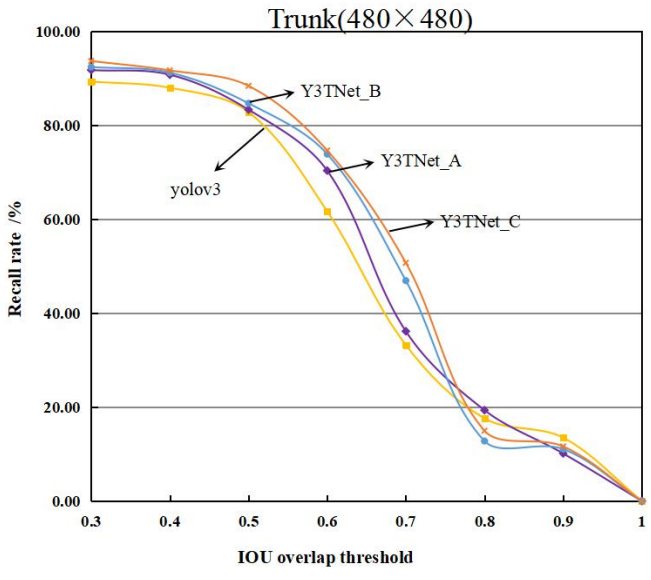


Fig.7. Testing results.
Note: The source image of Fig.7 (input size 480 × 480 pixels) is obtained by augmenting the image presented in Fig.5.

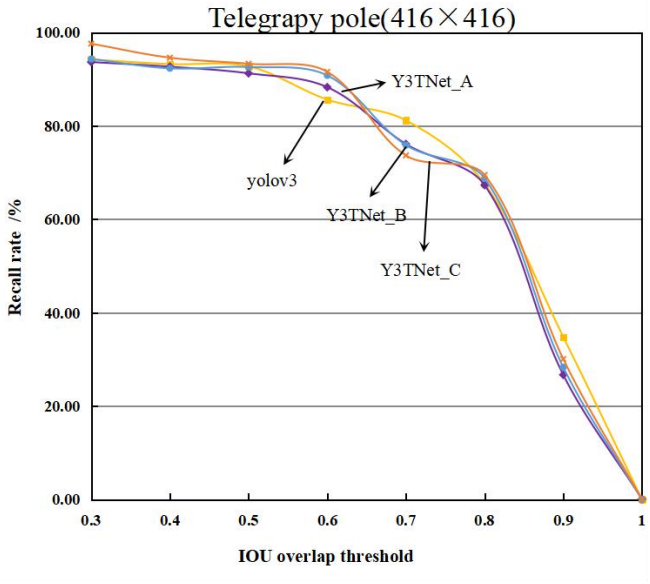
To evaluate the detection performance of models, we also compare the average recall rate of three types of detection objects in the trunk dataset for different IOU thresholds and input sizes as shown in Fig. 8. They (a_1 and a_2) shows the IOU_Recall curve with its gentle overall change. This figure better reflects the generalization ability of the models as compared with Fig. 6. Fig. 8(b_1 and b_2) and (c_1 and c_2) show the curves of the telephone pole and streetlight, respectively. The average recall rate of these two objects was better than that of the trunk under the same IOU value. By analyzing the trunk dataset, we found that the numbers of these two objects (telephone pole and streetlight) were relatively small, with a high certainty assigned to the marked object boundary boxes and with clear advanced features. For an IOU of 0.3, the average recall rate of the trunk, telephone pole, and streetlight for Y3TM_C (480 × 480 pixels) reached 93.60%, 98.61%, and 95.19%, respectively.



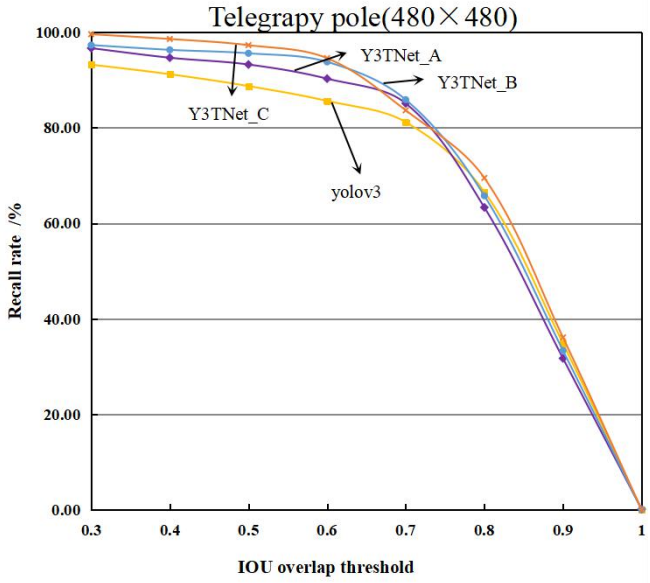
(a_1)



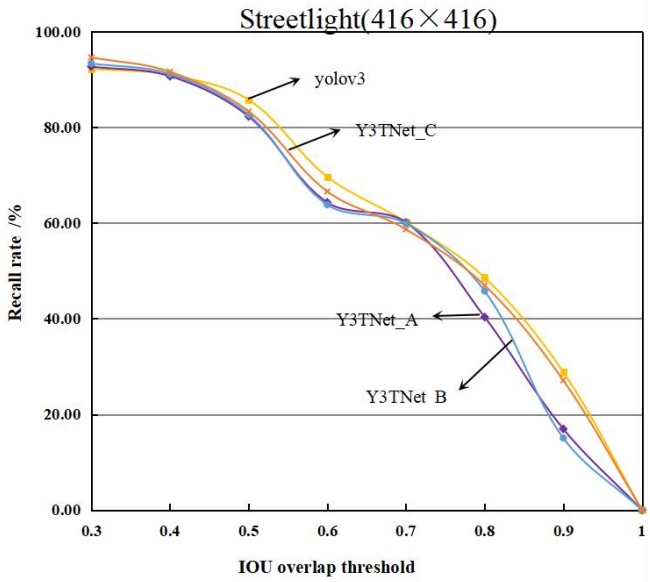
(a_2)



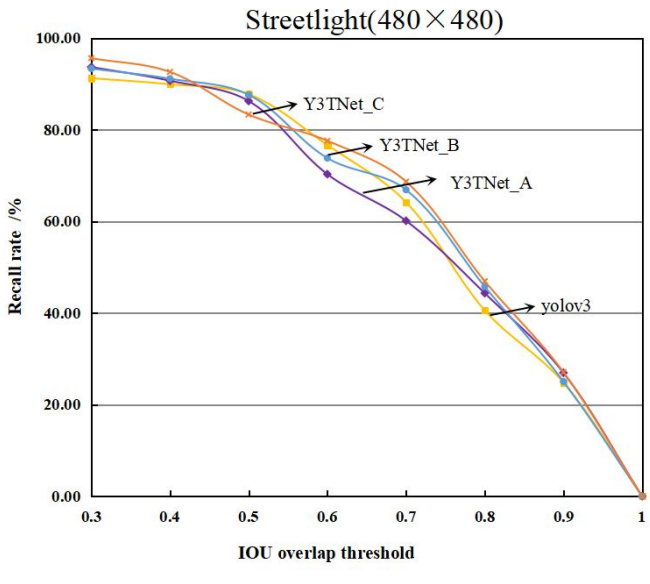
(b1)



(b2)



(c1)



(c2)

Fig.8. Comparison of IOU_Recall on trunk dataset

V. Conclusions and Future Works

Image-based objects detection, the trunk objects with distance from the camera in the image are small and difficult to detect. As we all know, the Yolo v3 algorithm can improve prediction accuracy while maintaining the speed advantage, especially strengthen the ability to recognize small objects. So, addressing the problems of traditional trunk detection methods, such as low accuracy, instability, and slow speed, we have proposed a rapid image detection method based on a convolutional neural network and transfer learning. We have designed a new trunk detection model called Y3TM to recognize multiple trunks in an image rapidly and accurately.

To address limitations in our dataset, we adopted data augmentation technology to improve the number and diversity of training samples. Using this dataset, we found that transfer learning-enabled faster convergence in our CNN. Customizing detection models with different input sizes and depths allowed us to better identify the object category and location in images although even the shallow network achieved great detection performance. Our Y3TM_C variant achieved a 93.60% average recall rate while our Y3TM_B (480×480) lowered the average error rate to 4.98%. Both variants had significantly shorter average detection times than yolov3.

However, our Y3TM model performed less well with images containing both near and distant objects. In the future, we plan to enrich the trunk dataset further to make full use of multi-scale features to establish establishing an end-to-end trunk detection model with excellent performance.

References

[1] X. Y. Chen, S. A. Wang, and B. Q. Zhang, "Multi-feature fusion tree trunk detection and orchard mobile robot localization using the camera/ultrasonic sensors," Computers and Electronics in Agriculture, vol. 147, pp. 91-108, 2018.

[2] S. Bargoti, J. P. Underwood, J. I. Nieto, and S. Sukkarieh, "A Pipeline for Trunk Detection in Trellis Structured Apple Orchards," Journal of Field Robotics, vol. 32, no 8, pp. 1075-1094.

[3] A. Kolb, C. Meaclem, X. Q. Chen, and R. Parker, "Tree trunk detection system using LiDAR for a semi-autonomous tree felling robot," in 2015 IEEE Conference on Industrial Electronics & Applications, Auckland, New Zealand, 2015.

[4] S. Lei, X. Chen, B. Milne, and G. Peng, “A novel tree trunk recognition approach for forestry harvesting robot,” in 2014 IEEE Conference on Industrial Electronics & Applications, Hangzhou, China, 2014.
[5] F. L. Guan, and A. J. Xu, “Tree DBH measurement method based on smartphone and machine vision technology,” Journal of Zhejiang A & F University, vol. 35, no. 05, pp. 892-899, 2018.
[6] P. L. Cheng, J. H. Liu, and D. Wang, “Measuring Diameters at Breast Height Using Combination method of Laser and Machine Vision,” Transactions of the Chinese Society for Agricultural Machinery, vol. 44, no. 11, pp. 271-275, 2013.
[7] M. A. Juman, Y. W. Wong, R. K. Rajkumar, and L. J. Goh, “ A novel tree trunk detection method for oil-palm plantation navigation,”. Computers & Electronics in Agriculture, vol. 128, pp. 172-180.
[8] G. N. Luo , S. Y. Dong, K. Q. Wang, W. M. Zuo, S. D. Cao, H. G. Zhang, ”Multi-views fusion CNN for left ventricular volumes estimation on cardiac MR images,”IEEE Transactions on Biomedical Engineering 65: 1924-1934, 2018.
[9] Qinghe Zheng, Mingqiang Yang, Xinyu Tian, Xiaochen Wang, and Deqiang Wang, "Rethinking the Role of Activation Functions in Deep Convolutional Neural Networks for Image Classification," Engineering Letters, vol. 28, no.1, pp80-92, 2020.
[10] Jie Li, Mingqiang Yang, Yupeng Liu, Yanyan Wang, Qinghe Zheng, and Deqiang Wang, "Dynamic Hand Gesture Recognition Using Multi-direction 3D Convolutional Neural Networks," Engineering Letters, vol. 27, no.3, pp490-500, 2019.
[11] Noritaka Shigei, Kazuki Mandai, Satoshi Sugimoto, Ryoma Takaesu, and Yoichi Ishizuka, "Land-Use Classification Using Convolutional Neural Network with Bagging and Reduced Categories," Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2019, 13-15 March, 2019, Hong Kong, pp7-11.
[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation,” in 2014 IEEE conference on computer vision & pattern recognition (CVPR), Washington, USA, 2014, pp. 580-587.
[13] K. M. He, X. Y. Zhang, S. Q. Ren, and J. Sun, “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition,”. IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 37, no. 9, pp. 1904-1916, 2014.
[14] S. Ren, K. He, R. Girshick, and S. Jian, “ Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” International conference on neural Information Processing Systems, Montreal, Canada, 2015, pp. 91-99.
[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection” in 2015 IEEE conference on computer vision & pattern recognition (CVPR), Boston, Massachusetts, 2015.
[16] J. Redmon, and A. Farhadi, “YOLO9000: Better, Faster, Stronger”. in 2017 IEEE conference on computer vision & pattern recognition (CVPR), Honolulu, HI, USA, 2017.
[17] [17]Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC (2016) SSD: Single Shot MultiBox Detector. In :2016 European Conference on Computer Vision (ECCV), pp 21-37.
[18] M. S. Long, C. J. Ouyang, H. Liu, and Q. Fu, “Image recognition of Camellia oleifera diseases based on convolutional neural network & transfer learning,” Transactions of the Chinese Society of Agricultural Engineering, vol. 34, no. 18, pp. 194－201, 2018.
[19] C. Deng, Y. X. Xue, X. L. Liu, C. Li, D. C. Tao,” Active Transfer Learning Network: A Unified Deep Joint Spectral–Spatial Feature Learning Model for Hyperspectral Image Classification,” IEEE Transactions on Geoscience and Remote Sensing, 57: 1741-1754, 2019.
[20] J. Redmon, and A. Farhadi, “YOLOv3: An Incremental Improvement”. in 2018 IEEE conference on computer vision & pattern recognition (CVPR), Salt Lake, Utah, 2018.
[21] Y. C. Zhou, T. Y. Xu, H. B. Deng, and T. Miao, “Real-time recognition of main organs in tomato based on channel wise group convolutional network,” Transactions of the Chinese Society of Agricultural Engineering, vol. 34, no. 10, pp. 153－162, 2018.
[22] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “ Rethinking the inception architecture for computer vision,” in 2016 IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 2818－2826.
[23] M. Everingham, L. V. Gool, and C. K. I Williams, “The pascal, visual object classes(VOC) challenge,” International Journal of Computer Vision, vol. 88, no 2, pp. 303－338, 2010.
[24] S. H. Qiu, G. J. Wen, Z. P. Deng, J. Liu, Y. X. Fan, “ Accurate non-maximum suppression for object detection in high-resolution remote sensing images,” Remote Sensing Letters,2018, 9: 238-247.

Ting-ting YANG /https://orcid.org/0000-0001-5441-1188 (0000-0001-5441-1188)
She received B.S degree in GIS from Anhui Science and Technology University in 2017. She is currently working as a Master' Degree Candidate of Zhejiang Agriculture and Forestry University in Zhejiang, China. Her main research covers Deep Learning and Image Processing.

Su-yin ZHOU /https://orcid.org/0000-0001-8501-3879
She was born in 1977 in Shandong Province, China, MA. Eng., Lecturer. She received the MA.Eng. degree in Control Theory and Control Engineering from Shandong University of Science and Technology in 2003. She is currently working as a Lecturer in the School of Information Engineering-Zhejiang Agriculture and Forestry University, Hangzhou, China. Her current research interest includes the application of the Internet of Things and Computer Application Technology in agriculture.

Ai-jun XU / https://orcid.org/0000-0001-6789-6938 (0000-0001-6789-6938)
He was born in 1976 in Anhui Province, China, Ph.D., Professor. He received the Ph.D. degree in Photogrammetry and Remote Sensing from Wuhan University in 2007. He is currently working as a Professor in the School of Information Engineering-Zhejiang Agriculture and Forestry University, Hangzhou, China. His current research interest includes Computer Application Technology and the application of GIS in the direction of agricultural informatization, etc.

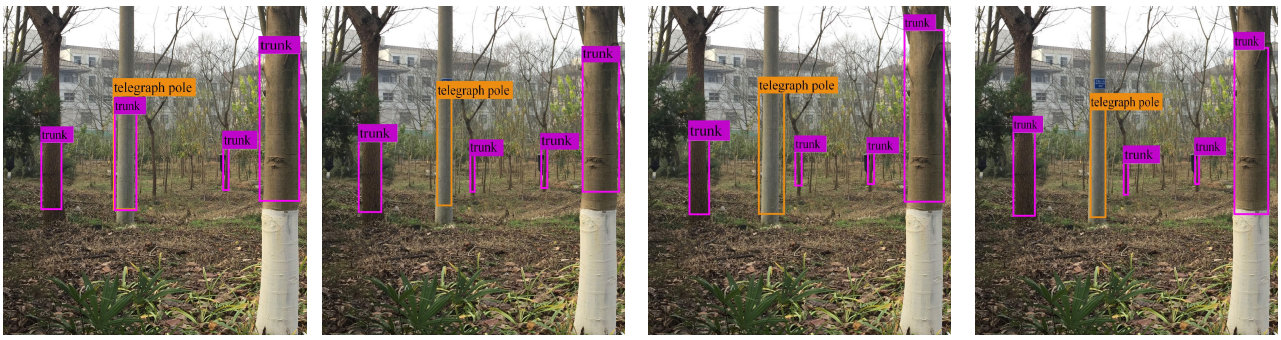
APPENDIX

yolov3 Y3TM_C

Fig. 4. Training results with different input sizes.

yolov3 Y3TM_A (480×480) Y3TM_B (480×480)

Fig. 5. Training results with different input sizes.



yolov3 Y3TM_A Y3TM_B Y3TM_C

Fig.7. Testing results.

Note: The source image of Fig.7 (input size 480 × 480 pixels) is obtained by augmenting the image presented in Fig.5.