Dense-YOLO: An Improved Weed Detection Platform Based on MSRCP

MingYuan Wang[1] and Watis Leelapatra[2]
[1,2]Department of Computer Engineering, Khon Kaen University, Thailand
E-mail: wangmingyuan@kkumail.com, watis@kku.ac.th

Received: August 24, 2022 / Revised: October 10, 2022 / Accepted: October 18, 2022

***Abstract*—For real-time weed detection needs and the flexibility of deploying models in embedded devices. We proposed an improved object detection platform, named Dense-YOLO which is based on Multi-scale retinex with chromaticity preservation (MSRCP) and YOLOv4 architecture. First, we use MSRCP to preprocess original images to provide a foundation for subsequent feature extraction. Second, Depthwise Separable Convolution (DSC) is used to reduce parameters, making it suitable for development on embedded devices. Third, we used k-means++ to optimize the clustering of anchor size. Fourth, DenseNet-121, PANet, and SPP modules together constitute Dense-YOLO. Last, we analyze the effectiveness of focal loss. Compared with YOLOv4, mAP is improved by 7.26%, three-quarters of the parameters are removed and 6.1 higher in FPS.**

***Index Terms*—Weed Detection, Neural Network, Lightweight Platform**

I. Introduction

Weeds are one of the biggest threats in agriculture, they compete with crops for water, sunshine, and nutrients, significantly affecting crop productivity and quality [1]. The traditional weeding methods are not suitable for modern agriculture, because of time-consuming, labor-intensive, and low efficiency. Mechanical, biological, thermal, electrical, and laser weeding technologies are rapidly attracting attention from scientists and research organizations [2]. Weed suppression is generally carried out at the seedling stage, and object detection is the first step before weeding [3].

In the research of weed detection, the wide applications of traditional computer vision mainly include two steps: feature extraction and pattern recognition [4]. For example, Pulido et al. [5] can improve the classifier's performance to above 90% based on Gray Levelco Occurrence Matrix (GLCM) and Support Vector Machines (SVM) [6]. Although this approach achieves high accuracy, it takes a long time to classify, has poor generalization, and is difficult to apply in practice.

Now, the emergence of neural networks and the popularity of deep learning have provided greater space for the development of object detection algorithms, such as LeNet [7], which was first used for document recognition. In recent years, many SOTA algorithms have been proposed, such as AlexNet [8], VGG [9], GoogLeNet [10], and ResNet [11]. Object detection algorithms have evolved from the traditional VJ Framework (used in face recognition) [12], HOG [13], and DPM (based on SVM) [14] to convolutional neural networks (CNNs), such as two-stage algorithms: SPPNet [15], Faster R-CNN [16] which is improved on Fast R-CNN [17], and one-stage algorithms: SSD [18]-[21], YOLO [22]-[25] that is a kind of end-to-end architecture and is famous as can balance with high accuracy and detection speed. Based on YOLOv3-tiny [24], Sharpe et al. [26] studied the influence of different annotation methods (entire plant annotation and partial annotation of leaf blade) and found the latter can have a better performance in detecting goosegrass. Gao et al. [27] also used YOLOv3-tiny as a foundation and presented an approach to improve the mean average precision (mAP) of convolvulus sepium and sugar beet to 82.9%. From the above research, we can achieve high accuracy through the Graphics Processing Unit (GPU). However, there are many embedded robots and devices that need lightweight algorithms that have low computational complexity and better detection performance during forward inference.

In this paper, Dense-YOLO: a novel lightweight weed detection platform is proposed, and it is based on YOLOv4 [25] architecture to detect garlic sprouts and weeds. During detection, it can trade-off between accuracy, model scale, and speed. Through experiments, we found that multi-scale retinex (MSR) [28] with chromaticity preservation (MSRCP) [29] can be a better choice to preprocess our original images because of its good performance. Meantime, we use k-means++ [30] cluster anchors instead of the original k-means method in our weed dataset can improve detection precision. DSC and Focal Loss [31] also be applied in our new network to be responsible for reducing the whole network's parameters and solving the class imbalance encountered during training,

respectively. Our proposed methods and Dense-YOLO have the characteristics of low parameters, high accuracy, small size, and good inference speed, that provide a new way of applications in embedded weeding robots.

II. Related Work

A. MSRCP

Land et al. [32] proposed the Retinex (consists of retina and cortex) theory that explained how the human visual system perceives a scene and divided the visual system model into two parts:

$$I(x,y) = L(x,y) \cdot R(x,y) \quad (1)$$

Where $I(x,y)$ denotes the final images received by eyes, and the low frequency information is denoted by $L(x,y)$ reflects the effect of lightness. $R(x,y)$ denotes scene reflectance and can reflect scene properties using high-frequency information. According to Retinex, Single-Scale Retinex (SSR), MSR, Multi-Scale Retinex with Color Restoration (MSRCR) and MSRCP were proposed one after another and be applied in image augmentation. While MSRCP can preserve better image chromaticity and color distribution by applying Retinex theory on intensive channels and mapping to each channel according to the original RGB scale. The intensity channel can be computed as follows:

$$I_{int} = (I_R + I_G + I_B)/3 \quad (2)$$

For those images with a correct color distribution and white lightning, MSRCP can avoid color distortion and restore better images while preserving the original color distribution. This provides an advantage for feature extraction in CNNs.

B. Depthwise Separable Convolution (DSC)

Different with standard convolution, DSC is constituted by depthwise convolution and pointwise convolution. As shown in

Fig. 1., the first step is to process the original $H \times W \times C$ images using $n \times n$ filters that number is same as input images, so the output is $(H - n + 1) \times (W - n + 1) \times C$ feature maps that will be the input images for the following phase. Second, pointwise convolution is in charge of mixing the information between channels. $1 \times 1 \times m$ filters will be used in this process, while m is the number of needed output feature maps. After both kinds of convolution, the final output feature maps shape is $(H - n + 1) \times (W - n + 1) \times m$.

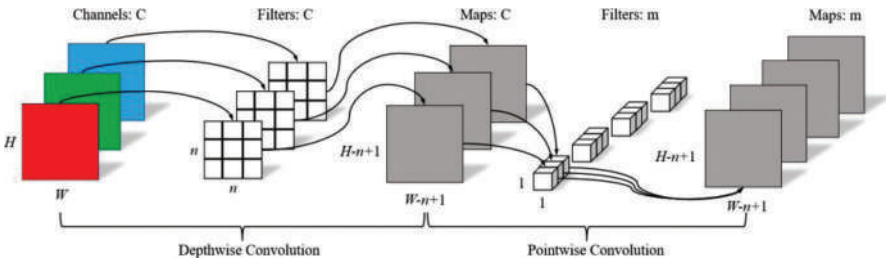

Fig. 1. Depth wise separable convolution process

The computational cost of standard convolution and DSC can be computed by Equations (3)-(4). Where, W_n and W_d denote the parameters of standard convolution and DSC respectively. W and H are the shapes of images, C is the number of channels. The size of the filter is represented by n. As shown in Equation (5), DSC has smaller parameters than standard convolution in the operation of convolution and is more suitable for deep learning networks.

$$W_n = m \times n^2 \times C \times (H - n + 1) \times (W - n + 1) \quad (3)$$

$$W_d = (n^2 + m) \times C \times (H - n + 1) \times (W - n + 1)$$

$$e = \frac{W_d}{W_n} = \frac{1}{m} + \frac{1}{n^2} \approx \frac{1}{n^2}, (m \gg n) \quad (4)$$

C. DenseNet

Dense convolutional network (DenseNet) [33] was proposed by Huang et al. in 2017 and is used in deeper convolution networks. The core point of DenseNet is to connect every layer to each other layer in a feed-forward fashion. As a traditional neural network that has L connections, there is $L \times (L + 1)/2$ direct connections between each layer and its subsequent layer in DenseNet. The feature maps of all preceding layers are used as inputs for each layer, and its own feature maps are used as inputs for all following layers. DenseNet not only can avoid vanishing gradient but also can reduce the scale of the network, and reuse and combine feature maps.

As illustrated in Fig. 2 each Dense Block contains enormous nodes that consist of normalization, activation functions, and convolutional layers, and are connected to the nodes before it. To increase inference efficiency, 1×1 convolution is used as a bottleneck layer to reduce the number of input feature maps.

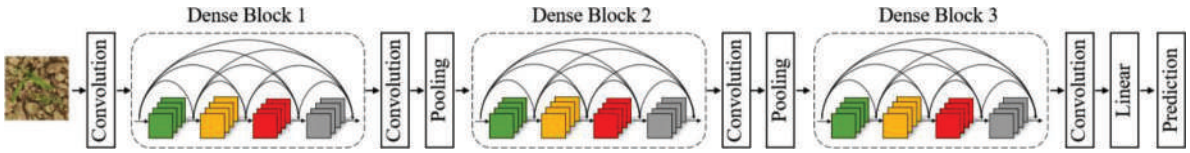

Fig. 2. DenseNet model structure

D. YOLOv4

From 2016-2018, Redmon et al. successively introduced YOLOv1 [22], YOLOv2 [23] and YOLOv3 [24] and realized object detection and classification prediction in a single network. This approach can detect objects rapidly while maintaining sufficient accuracy, which is suitable for real-world applications. Bochkovskiy et al. proposed YOLOv4 [25] in 2020, based on YOLOv3 and conducted some improvements to enhance feature extraction. Cross Stage Partial Darknet53 (CSPDarknet53), Spatial Pyramid Pooling (SPP), Path Aggregation Network (PANet), and YOLO Head comprise the YOLOv4 structure. Unlike YOLOv3, CSPDarknet53 is used instead of Darknet53, Mish activation replaces the LeakyReLu activation to improve the stability of gradient propagation. Three different size pooling layers in SPP can increase receptive field and fuse more feature maps. In PANet, multi-scale feature fusion occurred frequently to extract further features with various scales. After YOLO Head, we can get the anchor adjustive sizes, object location, classification confidence score and Intersection over Union (IoU) score. Finally, Non-Maximum Suppression (NMS) is used to filter badly bounding boxes. Equation (6) is to computed confidence score which reflect the credibility of predicted object.

$$C_i^j = P_r(Object) \times IoU_{pred}^{truth} \qquad (5)$$
$$P_r(Object) \in [0, 1]$$

Where, C_i^j denotes the confidence score of the jth bounding box in the ith grid. $P_r(Object)$ reflects the probability that one bounding box contains the target object.

from the advantages of YOLOv4 that can trade off detection accuracy and speed. Compared with the original YOLOv4, there are many improvement architectures proposed. For detecting apple flowers, based on YOLOv4, Wu et al. [34] used the channel pruning algorithm to achieve a huge reduction in model size, inference time, and network parameters. Dong et al. [35] focused on some attention methods to increase weed detection accuracy, such as shuffle attention and transformer module. Multi-scale fusion and pooling methods are also applied by Che et al. [36] to improve the dishes detection performance. According to different fields, YOLOv4 is improved and applied in their own research which is an inspiration for this paper.

III. Methodology

Fig. 3 is the proposed framework which is designed to achieve better performance in weed detection and has lower parameters in the whole network. Four techniques will be introduced in this framework: MSRCP is used as an enhancement method for original images. K-means++ can optimize the initial anchor size. Focal Loss aims to address the common issue of class imbalance. Dense-YOLO is an improved one-stage object detection based on DenseNet and YOLOv4. Their detail will be presented in the following sections.

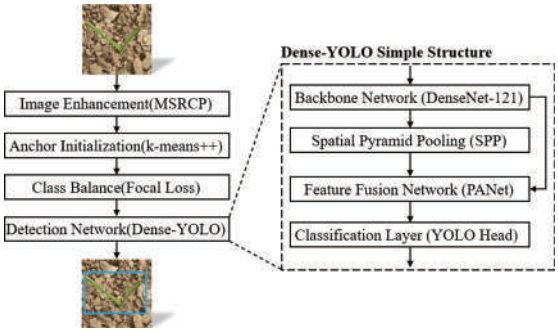

Fig. 3. The diagram of proposed framework

A. Image Enhancement

In the real world, those captured images have different lightness, dark regions, and poor contrast because of the effect of sunlight or the change of weather. Indeed, the result of applying data augmentation can restore a clearer and better image from an original image with dark regions or poor quality in other areas. In this paper, we tested different methods to find out the best one to support our next experiments, including Contrast Limited Adaptive Histogram Equalization (CLAHE) [37], homomorphic filtering based on Hue-Saturation-Intensity (HSI) [38], and MSRCP. CLAHE can reduce the noise problem and poor contrast by clipping several pixels that exceed the contrast factor to redistribute the whole histogram. According to the HSI of the original color image, the homomorphic filtering approach can not only improve the unequal distribution issue but also maintain the

RGB information. To preserve the original color distribution, MSRCP applies Retinex to the intensity channel without color restoration and maps the data to each channel using the original RGB scale. As shown in Fig. 4 comparative images were carried out in this paper to intuitively feel the impact of each algorithm on the original image. The contrast and color distribution after HFHSI processing have greatly improved. The lightness has increased using CLAHE, however, the contrast is relatively low. As for MSRCP, the leaves of the image are clearly distinguished from the background. Although the image is unnatural, the details are clearer, such as the classification texture, and the whole lightness is improved greatly, the saturation is better, and the contrast is enhanced.

As shown in Table I, we also conducted comparative tests based on YOLOv4 to find out the optimal image enhancement method that is suitable for the CNNs. MSRCP achieves a good performance both in vegetable and weed AP value and mAP is higher by 2.75% than the original image. So, we will use MSRCP as our image enhancement method for further experiments.

Fig. 4. The comparison of different image enhancement algorithm

TABLE I
COMPARISON OF DIFFERENT IMAGE ENHANCEMENT ALGORITHMS

Method	AP/%		Precision/%	Recall/%	mAP/%
	Vegetable	Weed			
Origin	90.70	79.81	92.87	82.64	85.26
HFHSI	91.13	76.03	92.22	83.14	83.58
CLAHE	90.48	75.90	94.08	82.30	83.19
MSRCP	93.07	82.95	92.43	87.65	**88.01**

B. Dimension Cluster

Instead of clustering anchors by standard k-means that are used in the original YOLOv4, we run k-means++ clustering on the training set bounding boxes to automatically find good anchors and avoid the shortcoming caused by random initial cluster center. Based on k-means++, a specific way of choosing anchors is used in this paper and we detail the whole process:

1) Prepare our bounding boxes dataset (all the class) and the number of needed centers k=9.

2) Take one random center from dataset N as the initial center O_1.

3) Take a new center O_x from dataset N, choosing x with $P(x)$.

$$D(x) = 1 \quad IoU(x, O_x) \tag{6}$$

$$P(x) = \frac{D(x)^2}{\sum_{x \in N} D(x)^2} \tag{7}$$

Where, $IoU(x, O_x)$ denotes IoU between data x and center O_x. $D(x)$ is the distance of data x with its adjacent center O_x. $P(x)$ is the probability of being the next center.

4) Repeat step 3, until we have taken 9 centers altogether.

5) Iterate compute $D(x)$, $x \in N$, classify $x \in D(x)_{\min}$ to the new cluster center C_i that can get by Equation (9).

$$C_i = \frac{\sum_{x \in C_i} x}{|C_i|} \tag{8}$$

6) When the value of C_i is stable, the output is the optimal cluster results.

We implemented the above algorithm and conducted combined experiments on our bounding boxes dataset and get the best performance anchors group: (130, 124), (83, 236), (195, 174), (305, 197), (259, 266), (308, 322), (570, 814), (1764, 449), and (2370, 1268).

C. Dense-YOLO

Based on YOLOv4, we proposed a novel end-to-end network: Dense-YOLO. Instead of using CSPDark53 as the backbone, we introduced DenseNet-121 as the backbone to rich the feature extraction and stable gradient propagation, as shown in Fig. 5. Through SPP structure, maxpooling layers of different sizes can extract more features from backbone network and increase receptive field. The purpose of the transition section which consists of 1×1 convolution and average pool is to change the number of channels and condense feature maps.

After transition section, we need to bridge the backbone and PANet, feature maps with different shapes are divided to three parts: 52×52×256, 26×26×512, and 13×13×1024. As like the original YOLOv4, those feature maps are responsible for detecting different size objects respectively. To lightweight our network, we replace all the 3×3 standard convolutions with DSC, Finally the number of parameters is decreased to one-quarter of the original YOLOv4. This strategy increases the feasibility of the object detection method being deployed on embedded robots or devices. In the improved backbone, like DenseNet-121, four Dense Blocks are used to improve the learning ability of feature extraction. The number of Dense units in each Dense Block is 6, 12, 24, and 16 respectively. ResNets and DenseNet convolutional feed-forward networks layer transition are shown by Equations (10) and (11). X_l is the output of l^{th} layer. $H(\cdot)$ denotes the function of operations such as convolution, Batch Normalization (BN) [39], rectified linear activation (ReLU) [40], or pooling. Although they all belong to residual networks, DenseNet combines the feature maps of all preceding layers and can reuse features throughout the networks to obtain a more accurate model.

$$X_l = H_l(X_{l-1}) + X_{l-1} \tag{9}$$

$$X_l = H_l([X_0, X_1, \cdots, X_{l-1}]) \tag{10}$$

In the part of feature fusion, Dense-YOLO method keeps the PANet structure to extract feature maps with different scales to increase object detection speed, as the same YOLOv4 method. In the end, three head parts are used to predict classes and bounding boxes of weeds and vegetables.

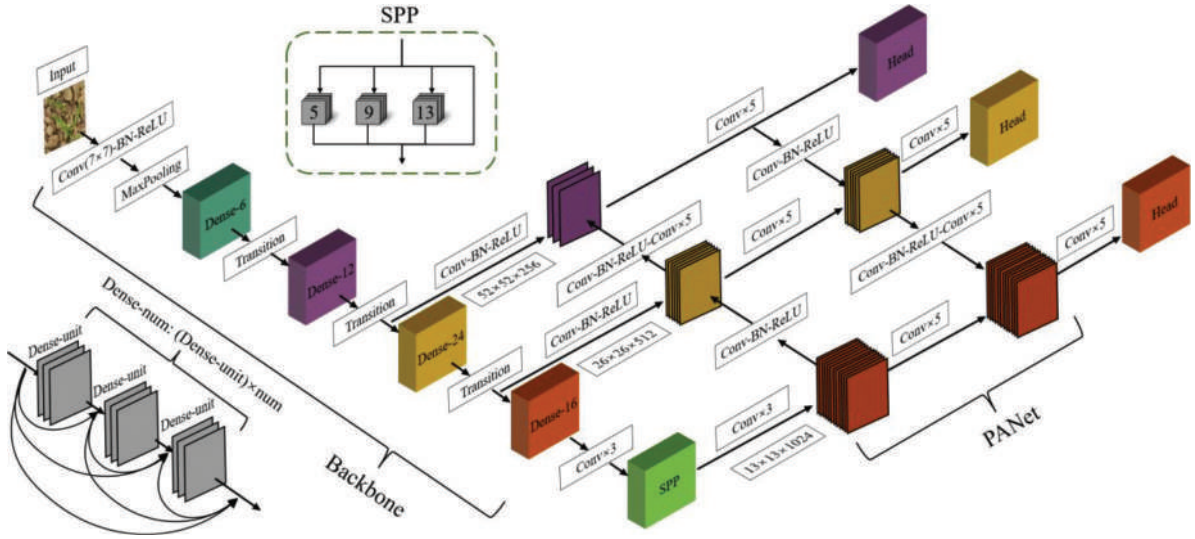

Fig. 5. Dense-YOLO model structure

D. Class Balance

Actually, class imbalance is a common phenomenon, and there are a lot of vegetables and small weeds in the real dataset. Using standard cross entropy as a loss function may cause iteration to slow or deviate from the correct direction when training on the dataset with foreground-background class imbalance. So, we introduce Focal Loss in our model to make the network focus on hard, misclassified examples during training. Based on standard cross entropy, Focal Loss introduced a weighting factor $\alpha \in [0, 1]$ to address class imbalance as shown in Equation (12).

$$Loss_{Focal} = -\alpha(-P(y|x))^{\gamma} \log(P(y|x)) \tag{11}$$

Where α for the positive class and $1-\alpha$ for the negative class. $\gamma \geq 0$ is focusing parameter, it can adjust the loss speed of easy examples and address the network to focus on the hard negatives. $P(y|x) \in [0, 1]$ denotes the estimated probability for the example x with label y in our model. We implement the Focal Loss method to Dense-YOLO to limit the dominant role of easy examples, avoid learning useless features, and improve training efficiency.

IV. Experiments

A. Dataset

A large-scale dataset is necessary for supervised learning and deep learning. That's why we established our own dataset for weed detection through capturing images from farmland at a height of 40 cm. We capture images during stem elongation stage because there are more features that can be distinguished, and occlusion has less impact on each object. Our dataset contains 1800 original images and two categories: vegetable and weed. In the real world, our dataset has following properties: (a) occlusion with each other, (b) incomplete objects on the edge of image, (c) other objects coexist with our target object, and (d) dark regions, as shown in Fig. 6.

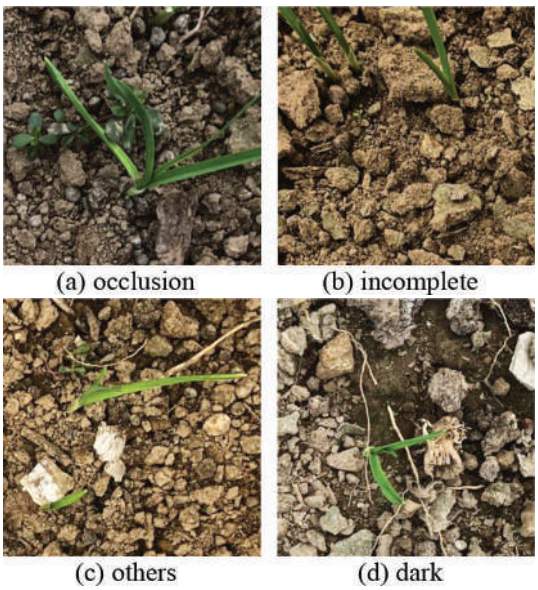

Fig. 6. Example images of vegetable and weed

Data augmentation is widely used in State-of-the-Art (SOTA) detection algorithms and can boost a network's performance and robustness. We add rotation, flipping, scaling, blurring, and cropping into augmentation strategies to expand image number and prevent overfitting. As shown in Table II, the original images are expanded to 5000, and 90% of them are used as training and validation, and 10% are to test performance.

TABLE II
Statistical Data of Augmentation

Origin	Augmented	Train	Validation	Test
1800	5000	4050	450	500

B. Evaluation

Popular measures used to evaluate detection performance include Precision (P), Recall (R), Average Precision (AP), mean mAP, Frame Per Second (FPS), and the number of parameters in the whole network. Using Equation (13) to calculate the P and Equation (14) to get R which is the proportion of predicted items to the total number of true items. The P-R curve can be obtained by P and R, and the area enclosed by the P-R curve with the x and y axes is AP. mAP is the value by summing the AP of each category and then averaging.

$$P = \frac{TP}{TP + FP} \tag{12}$$

$$R = \frac{TP}{TP + FN} \tag{13}$$

$$mAP = \frac{\sum_{i=1}^{k} AP_i}{k} \tag{14}$$

Where TP denotes the positive examples are correctly identified as positive samples. FP is the negative examples that are incorrectly identified as positive samples. FN indexes the positive samples are incorrectly identified as negative samples. k is the number of detection classes.

C. Clustering Analysis

Dense-YOLO is an anchor-based detector, anchors with suitable size are the crucial factor for yielding good result performance. The size distribution of bounding boxes differs from other common datasets, so it is necessary to cluster before training. Instead of k-means method, we use k-means++ to conduct clustering analysis in each class. We set k as 9 because Small (S), Medium (M), and Large (L) are three groups for detecting different-sized objects and each group has three pairs of data. Table III presents the results. Moreover, to study the effects of different group combinations, mixed experiments are conducted to find a better anchor setting. Table IV displays group settings and comparison results. In total, there are 9 groups from A to I.

Table V is our experimental setup: all training and tests are conducted with 3060Ti GPU and use this GPU to execute multi-scale training in the batch size of 64 while mini-batch size is 8 or 4 depending on GPU memory limitation. This work uses Python 3.6 and Pytorch1.10 as the programming language and deep learning platform. During training, the Adam method is the optimizer to implement gradient backpropagation. The total epoch, momentum, learning rate, and decay are 200, 0.937, 0.001, and 0.0005 for all methods, respectively.

TABLE III
EXPERIMENTAL RESULTS OF K-MEANS++

	Weed			Vegetable		
	S	**M**	**L**	**S**	**M**	**L**
Clustering by Category	130,124	305,197	271,457	150,142	506,234	570,814
	83,236	259,266	546,240	287,197	534,466	1764,449
	195,174	308,322	419,348	252,420	895,389	2370,1268
Together Clustering	(148,140) (280,208) (259,421) (498,238) (534,468) (895,389) (570,814) (1707, 421) (2265,1256)					

TABLE IV
MIXED COMPARISON EXPERIMENTS GROUPS SETTING AND DETECTION RESULTS

Combination Setting							**Experimental Results**				
	Weed			**Vegetable**			**Weed**	**Vegetable**	**Precision/%**	**Recall/%**	**mAP/%**
Group	**S**	**M**	**L**	**S**	**M**	**L**	**AP/%**				
A	**Together Clustering Result**						78.86	93.07	93.66	88.81	85.97
B	√	-	-	-	√	√	80.00	92.70	93.83	88.81	86.35
C	-	√	-	√	-	√	81.76	93.82	93.90	89.98	87.76
D	-	-	√	√	√	-	84.10	94.08	93.15	90.82	89.09
E	√	√	-	-	-	√	85.41	93.72	94.07	89.98	**89.56**
F	-	√	√	√	-	-	84.69	92.70	91.79	89.65	88.69
G	√	-	√	-	√	-	79.38	92.70	93.51	88.98	86.04
H	√	√	√	-	-	-	79.35	93.40	94.37	89.48	86.37
I	-	-	-	√	√	√	84.63	93.91	93.16	90.15	89.27

TABLE V
THE HARDWARE AND DETAILS OF MODEL TRAINING EXPERIMENTAL SETUP

Setup	**GPU**	**Programming Language**	**Platform**	**Optimizer**	**Epoch**	**Momentum**	**Learning Rate**	**Decay**
Details	3060Ti	Python 3.6	Pytorch 1.10	Adam	200	0.937	0.001	0.0005

Based on YOLOv4, we choose MSRCP algorithm to process original images before feeding into the neural network. From Table IV, different anchors combination does affect the result and group E achieves the best performance with 89.56% mAP. Compared with the original k-means method, The relative mAP has improved by 1.55%. The reason is the suitable combination makes it easy for detector to adjust bounding boxes and finish regression. Next, we will use the anchor size of group E to continue experiments.

D. Lightweight Network

To reduce the total parameters and increase interference speed during detection, Howard et al. proposed MobileNetV1 [41], MobileNetV2 [42] and MobileNetV3 [43]. Because DSC was introduced into their network to reduce parameters, while trade-off between accuracy, model size, and detection speed. In this paper, we also use DSC to replace the 3×3 standard convolution to lightweight our network and make it possible for applying to embedded devices. Additionally, we used DSC in YOLOv4 rather than traditional convolution to test the efficacy of DSC.

TABLE VI
COMPARISON EXPERIMENTS

Convolution Method	AP/%		Precision/%	Recall/%	mAP/%	Time/ms	FPS	Model Size/MB	Parameter
	Weed	Vegetable							
Standard Convolution	85.41	93.72	94.07	89.98	89.56	56.9	17.6	244	64,009,375
DSC	75.67	91.05	92.76	87.65	83.36	44.7	22.4	137	35,757,727

TABLE VII
COMPARISON OF DIFFERENT BACKBONES

Backbone	AP/%		Precision/%	Recall/%	mAP/%	Time/ms	FPS	Model Size/MB	Parameter
	Weed	Vegetable							
CSPDarkNet53	75.67	91.05	92.76	87.65	83.36	44.70	22.40	137.00	35,757,727
MobileNetV1	75.46	94.09	94.17	88.98	84.77	19.90	50.30	51.10	12,271,999
MobileNetV2	74.12	91.45	93.31	88.48	82.78	22.20	45.00	46.50	10,381,119
MobileNetV3	73.10	88.49	90.78	85.48	80.80	24.10	41.50	53.70	11,309,039
GhostNet	74.78	92.24	92.91	87.48	83.51	31.70	31.50	42.50	11,008,515
ResNet-50	83.81	94.43	92.84	90.98	89.12	36.80	27.20	127.00	33,261,183
VGG-16	86.15	95.80	94.23	92.65	90.98	63.00	15.90	89.90	23,517,567
DenseNet-121	84.67	95.06	93.52	91.49	89.86	41.50	24.10	61.90	16,018,879

As shown in Table VI, DSC reduces half of the parameters and computational cost in the whole network, but the mAP also has a big drop at the same time. Don't worry, this will be improved through our proposed Dense-YOLO which is also based on DSC.

E. Backbone Improvement

To slim our network and keep high accuracy, we evaluate and compare the performances of different architectures as feature backbone. Besides MobileNets, GhostNet [44], ResNet-50, VGG-16 and DenseNet-121 also participated in the comparison experiments. There are three major observations from Table VII:

1) The situation is changed with different backbone-we can further reduce the total parameters by combining different feature extraction backbone. This indicates that the model size problem is well addressed in this operation, and we manage to obtain more accuracy gains from modified backbone.

2) We also note that the number of parameters is not the decisive factor for precision, recall and mAP. A new network that trade-off accuracy and model size can be obtained from a suitable backbone. More importantly, the detection speed or FPS increased along with the reduction of parameters, this is significant for real-time models.

3) Compared to the original YOLOv4 backbone (CSPDarkNet53), VGG-16 achieves the top-1 mAP 90.98%, however other measures are not ideal. While DenseNet-121 not only performs better with 89.86 mAP, but also has a lower parameter and faster detection speed. Because of this, we use DenseNet-121 as the backbone of Dense-YOLO, so that the network parameters can be dramatically reduced to a quarter of the YOLOv4, while the FPS is increased by 6.5, and the mAP doesn't drop down.

F. Focal Loss Test

As we discuss in class balance section, here we introduce Focal Loss into Dense-YOLO to see its effect during training. Table VIII is the results before and after using Focal Loss. This comparison successfully verifies the effectiveness of Focal Loss on imbalance dataset, we achieve 92.52% of mAP, a 2.66% improvement.

TABLE VIII
COMPARISON OF USING FOCAL LOSS AND NOT

	AP/%		Precision	Recall	mAP
	Weed	Vegetable			
Traditional	84.67	95.06	93.52	91.49	89.86
Focal Loss	96.05	88.99	96.10	90.48	**92.52**

G. Comparison with SOTA

To demonstrate the superiority of Dense-YOLO, comparisons are made with other state-of-the-arts. We compared detection algorithms with different structures such as SSD, CenterNet [45], EfficientDet [46], YOLOv4-Tiny [47] and RetinaNet since we want to design another excellent structure. As shown in Table IX. Tests show that Dense-YOLO can substantially reduce model size and increase speed while maintaining the stability of mAP compared to original YOLOv4. YOLOv4-Tiny and Efficient Det-D0 can minimize model size and improve FPS, unfortunately, they also cause a huge decrease in mAP. Surprisingly, Dense-YOLO is 26.3% higher

in mAP than YOLOv4-Tiny, and compared to original YOLOv4, it is not only 7.26% higher in mAP, but also three-quarters of the parameters are removed and 6.1 higher in FPS.

Why this proposed model can get the best result? The reasons can be attributed to the techniques we discussed above. MSRCP improves the quality and feature distribution compared to the original images. K-means++ makes the initial anchor size closer to most of the target objects. Class imbalance can be weakened through Focus Loss, thus the mAP has a further increment. Meanwhile, DenseNet-121 can deepen layers of the backbone network to fit the more complex model. While the use of DSC reduces the compute parameters and model size.

TABLE IX
COMPARISON WITH OTHER STATE-OF-THE-ARTS

Model	AP/%		mAP/%	Time/ms	FPS	Model Size/MB	Parameter
	Weed	Vegetable					
YOLOv4	90.70	79.81	85.26	54.30	18.40	244.00	64,009,375
YOLOv4-Tiny	52.26	80.19	66.22	8.40	118.50	22.50	5,882,634
SSD	58.73	84.44	71.59	26.10	38.40	91.20	23,745,908
Center Net	69.90	89.16	79.53	32.60	30.70	125.00	32,718,597
Efficient Det-D0	75.18	92.51	83.85	36.40	27.50	16.10	3,875,723
Retina Net	61.23	82.78	72.00	86.40	11.60	138.00	36,350,582
Dense-YOLO	96.05	88.99	**92.52**	40.70	24.50	61.90	16,018,879

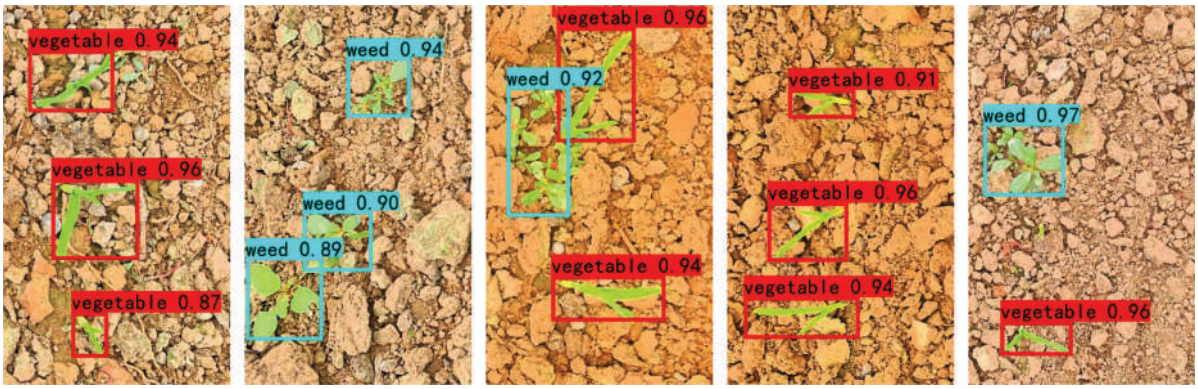

Fig. 7. Detection results from Dense-YOLO

V. CONCLUSION

For the needs of weeding robots or other embedded devices, this paper proposed a compacted, lightweight and speed detection method in terms of network structure, we term it Dense-YOLO. MSRCP algorithm is introduced to preprocess images to increase color contrast and remove dark regions. To lightweight the whole network, we replaced all 3×3 standard convolution with DSC and achieved a noticeable effect in the parameter. We used k-means++ algorithm to optimize the clustering of anchor size, instead of the traditional method. After extensive comparison analysis, we used DenseNet-121, PANet and SPP modules together to constitute the network of Dense-YOLO, a plug-and-play model for smart agriculture and is a portable neural architecture to promote weed detection in both flexibility and accuracy. In the end, we also demonstrated the effectiveness of Focal Loss in solving the class imbalance problem. We are pleased to present our encouraging results and will continue to focus on computer vision and neural network research in future work, such as the network pruning to further slim our model.

REFERENCES

[1] D. D. Patel and B. A. Kumbhar, "Weed and its Management: A Major Threats to Crop Economy," *J. Pharm. Sci. Biosci. Res*, vol. 6, pp. 453-758, 2016.

[2] L. Wen, X. Liming, and X. Jiejie, "Research Status of Mechanical Intra-Row Weed Control in Row Crops," *Journal of Agricultural Mechanization Research*, vol. 39, pp. 243-250, 2017.

[3] B. Liu and R. Bruch, "Weed Detection for Selective Spraying: A Review," *Current Robotics Reports*, vol. 1, pp. 19-26, Mar. 2020.

[4] S. Shanmugam, E. Assunção, R. Mesquita et al., "Automated Weed Detection Systems: A Review," *KnE Engineering*, pp. 271-284, Jun. 2020.

[5] C. Pulido, L. Solaque, and N. Velasco, "Weed Recognition by SVM Texture Feature Classification in Outdoor Vegetable Crop Images," *Ingeniería e Investigación*, vol. 37, pp. 68-74, Apr. 2017.

[6] S. R. Gunn, "Support Vector Machines for Classification and Regression," *ISIS Technical Report*, vol. 14, pp. 5-16, May. 1998.
[7] Y. Lecun, L. Bottou, Y. Bengio et al., "Gradient-Based Learning Applied to Document Recognition," in *Proc. IEEE*, pp. 2278-2324, 1998.
[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Communications of the ACM*, vol. 60, pp. 84-90, May. 2017.
[9] K. Simonyan and A. Zisserman. (2014, Sep. 4). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. [Online]. Available: https://arxiv.org/abs/1409.1556
[10] C. Szegedy, W. Liu, Y. Jia et al., "Going Deeper with Convolutions," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1-9.
[11] K. He, X. Zhang, S. Ren et al., "Deep Residual Learning for Image Recognition," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770-778.
[12] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision*, vol. 57, pp. 137-154, May. 2004.
[13] N. Dalal and B. Triggs, "Histograms of Oriented Gradients for Human Detection," in *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005, pp. 886-893.
[14] P. F. Felzenszwalb, R. B. Girshick, D. McAllester et al., "Object Detection with Discriminatively Trained Part-Based Models," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1627-1645, Sep. 2010.
[15] K. He, X. Zhang, S. Ren et al., "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, pp. 1904-1916, Jan. 2015.
[16] S. Ren, K. He, R. Girshick et al., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *Advances in Neural Information Processing Systems*, Jun. 2015.
[17] R. Girshick, "Fast R-Cnn," in *Proc. IEEE International Conference on Computer Vision*, 2015, pp. 1440-1448.
[18] C. Fu, W. Liu, A. Ranga et al. (2017, Jan. 23). *DSSD: Deconvolutional Single Shot Detector*. [Online]. Available: https://arxiv.org/abs/1701.06659
[19] Z. Li and F. Zhou. (2017, Dec. 4). *FSSD: Feature Fusion Single Shot Multibox Detector*. [Online]. Available: https://arxiv.org/abs/1712.00960
[20] J. Jeong, H. Park, and N. Kwak. (2017, May. 26). *Enhancement of SSD by Concatenating Feature Maps for Object Detection*. [Online]. Available: https://arxiv.org/abs/1705.09587
[21] W. Liu, D. Anguelov, D. Erhan et al., "SSD: Single Shot Multibox Detector," *European Conference on Computer Vision*, pp. 21-37, Otc. 2016.
[22] J. Redmon, S. Divvala, R. Girshick et al., "You Only Look Once: Unified, Real-Time Object Detection," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779-788.
[23] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7263-7271.
[24] J. Redmon and A. Farhadi. (2018, Apr. 6). *YOLOv3: An Incremental Improvement*. [Online]. Available: https://arxiv.org/abs/1804.02767
[25] A. Bochkovskiy, C. Wang, and H. M. Liao. (2020, Apr. 23). *YOLOv4: Optimal Speed and Accuracy of Object Detection*. [Online]. Available: https://arxiv.org/abs/2004.10934
[26] S. M. Sharpe, A. W. Schumann, and N. S. Boyd, "Goosegrass Detection in Strawberry and Tomato Using a Convolutional Neural Network," *Scientific Reports*, vol. 10, pp. 1-8, Jun. 2020.
[27] J. Gao, A. P. French, M. P. Pound et al., "Deep Convolutional Neural Networks for Image-Based Convolvulus Sepium Detection in Sugar Beet Fields," *Plant Methods*, vol. 16, pp. 1-12, Dec. 2020.
[28] Z. Rahman, D. J. Jobson, and G. A. Woodell, "Multi-Scale Retinex for color image enhancement," in *Proc. 3rd IEEE International Conference on Image Processing*, 1996, pp. 1003-1006.
[29] A. B. Petro, C. Sbert, and J. Morel, "Multiscale Retinex," *Image Processing on Line*, vol. 4, pp. 71-88, Apr. 2014.
[30] D. Arthur and S. Vassilvitskii, "k-means++: The Advantages of Careful Seeding," in *Proc. Symp. Discrete Algorithms*, 2007, pp. 1027-1035.
[31] T. Y. Lin, P. Goyal, R. Girshick et al., "Focal Loss for Dense Object Detection," in *Proc. IEEE International Conference on Computer Vision*, 2017, pp. 318-327.
[32] E. H. Land, and J. J. McCann, "Lightness and Retinex Theory," *Journal of the Optical Society of America*, vol. 61, no. 1, pp. 1-11, Jan. 1971.
[33] G. Huang, Z. Liu, L. Van Der Maaten et al., "Densely Connected Convolutional Networks," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4700-4708.
[34] D. Wu, S. Lv, M. Jiang et al., "Using Channel Pruning-Based YOLO v4 Deep Learning Algorithm for the Real-Time and Accurate Detection of Apple Flowers in Natural Environments," *Computers and Electronics in Agriculture*, vol. 178, p. 105742, Nov. 2020.
[35] H. Dong, X. Chen, H. Sun et al., "Weed Detection in Vegetable Field Based on Improved YOLOv4 and Image Processing," *Journal of Graphics*, vol. 43, pp. 559-569, Mar. 2022.
[36] X. Che and H. Chen, "Muti-Object Dishes Detection Algorithm Based on Improved YOLOv4," *Journal of Jilin University*, pp. 1-7, Nov. 2021.
[37] K. Zuiderveld, "Contrast Limited Adaptive Histogram Equalization," *Graphics Gems*, pp. 474-485, May. 1994.
[38] Y. Zhang and M. Xie, "Colour Image Enhancement Algorithm Based on HSI and Local Homomorphic Filtering," *Computer Applications and Software*, vol. 30, pp. 303-307, Dec. 2013.
[39] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," in *Proc. International Conference on Machine Learning*, 2015, pp. 448-456.
[40] X. Glorot, A. Bordes, and Y. Bengio, "Deep Sparse Rectifier Neural Networks," in *Proc. the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011, pp. 315-323.
[41] A. G. Howard, M. Zhu, B. Chen et al. (2017, Apr. 17). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. [Online]. Available: https://arxiv.org/abs/1704.04861
[42] M. Sandler, A. Howard, M. Zhu et al., "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510-4520.
[43] A. Howard, M. Sandler, G. Chu, et al., "Searching for MobileNetV3," in *Proc. IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1314-1324.
[44] K. Han, Y. Wang, Q. Tian et al., "GhostNet: More Features from Cheap Operations," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1580-1589.
[45] X. Zhou, D. Wang, and P. Krähenbühl. (2019, Apr. 16). *Objects as Points*. [Online]. Available: https://arxiv.org/abs/1904.07850
[46] M. Tan, R. Pang and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10781-10790.
[47] Z. Jiang, L. Zhao, S. Li et al., (2020, Nov. 9). *Real-Time Object Detection Method Based on Improved YOLOv4-Tiny*. [Online]. Available: https://arxiv.org/abs/2011.04244

MingYuan Wang received the B.Eng. degree in Mechanical Design, Manufacturing and Automation from Henan University of Science and Technology, Henan, China, in 2013. Past work experience on acting product designer at YTO Group Corporation from 2014 to 2020. He current is a graduate student at Department of Computer Engineering, Khon Kaen University, Thailand. His current research interests include computer vision and its applications to other science and engineering fields.

Watis Leelapatra received the B.Eng in Computer Engineering from Khon Kaen University, Khon Kaen, Thailand, and the M.S. degree in Computer Science from the Case Western Reserve University, Cleveland, USA, and the D.Eng. Degree in Computer Science from Asian Institute of Technology, respectively. He current is a lecturer at the Department of Computer Engineering, Khon Kaen University.