FSD: feature skyscraper detector for stem end and blossom end of navel orange

Xiaoye Sun¹,² · Gongyan Li¹ · Shaoyun Xu¹

Received: 12 November 2019 / Revised: 24 September 2020 / Accepted: 7 October 2020 / Published online: 24 October 2020

Abstract
To accurately and efficiently distinguish the stem end and the blossom end of a navel orange from its black spots, we propose a feature skyscraper detector (FSD) with low computational cost, compact architecture and high detection accuracy. The main part of the detector is inspired from small object that the stem (blossom) end is complex and the black spot is densely distributed, so we design the feature skyscraper networks (FSN) based on dense connectivity. In particular, FSN is distinguished from regular feature pyramids, and which provides more intensive detection of high-level features. Then we design the backbone of the FSD based on attention mechanism and dense block for better feature extraction to the FSN. In addition, the architecture of the detector is also added Swish to further improve the accuracy. And we create a dataset in Pascal VOC format annotated three types of detection targets the stem end, the blossom end and the black spot. Experimental results on our orange dataset confirm that the FSD has competitive results to the state-of-the-art one-stage detectors like SSD, DSOD, YOLOv2, YOLOv3, RFB and FSSD, and it achieves 87.479% mAP at 131 FPS with only 5.812M parameters.

Keywords Real-time small object detection · Convolutional neural network · Feature skyscraper · Navel orange

1 Introduction
The surface defect of navel orange can be easily detected in traditional image processing, but the stem end and the blossom end of the navel orange are also drastically mistaken as defect. This may drive up economic losses. Despite the four types of symptoms hard spot, false melanoses or speckled blotch, freckle spot, and virulent or spreading spot commonly appear as black spots or blotches [1], we still only focus on the visual black spot, which is easier to confuse with a stem end and a blossom end. With the breakthrough progress in deep learning in image processing in recent years, that it surpasses the performance of traditional image processing [2] allows us to apply this technique to reduce the false positive rate.

Most of the models for detection tasks today are based on public datasets, like ImageNet, MS COCO, Pascal VOC, CIFAR-10, etc. Their good versatility often requires a more complex architecture. For some specific datasets with few detection categories and high homogeneity, although the models may have good performance after fine tuning, such architectures are very redundant and take up too much computational cost. Especially for the navel orange detection task, the dataset is relatively simple and has a strong distribution law. Therefore, we consider whether we can design and optimize a network architecture based on the statistical characteristics of the dataset, so that the network architecture can achieve high performance with the most compact structure possible. Indeed, our model FSD shows good performance as shown in Fig. 1.

Fine-tuning restricts the design of the model architecture and may cause learning bias and domain mismatch problems [3], so we adopt the dense connectivity proposed in DenseNet [4]. This strategy makes training from scratch in DSOD [3] possible.

The architecture of the FSD consists of three parts: a backbone, FSN and detection layers. Different from the regular
Fig. 1 FSD delivers the best performance with the most compact architecture and faster speed in our navel orange dataset. The size of the circle represents the size of the model.

For the backbone, it consists of three parts: a stem, a dense block and a transition layer. The dense block is similar to FSN and is designed by dense connectivity. The transition layer connects the dense block and the feature skyscraper network. We add an Squeeze-and-Excitation (SE) layer [5] to both the stem and transition modules. And all activation functions in the detection of the multi-scale features on the resolutions of the feature maps and achieves the multi-scale features by presetting the default boxes. The resolution of each layer of the skyscraper remains the same so that the higher level can design smaller default boxes. However, the higher levels of the general feature pyramids are invalid for small objects. This designed structure ensures the invariance of the resolution of the feature maps and avoids the problem that the low-resolution feature map limits the design of the minimum default box. Furthermore, the default box corresponding to the feature map can effectively cover the detection area at each scale, and larger detection targets also have denser default boxes. The size and aspect ratio of the default boxes corresponding to the feature maps are determined by the bounding boxes of clustering our navel orange dataset.

For the backbone, it consists of three parts: a stem, a dense block and a transition layer. The dense block is similar to FSN and is designed by dense connectivity. The transition layer connects the dense block and the feature skyscraper network. We add an Squeeze-and-Excitation (SE) layer [5] to both the stem and transition modules. And all activation functions in the detection of the multi-scale features on the resolutions of the feature maps and achieves the multi-scale features by presetting the default boxes. The resolution of each layer of the skyscraper remains the same so that the higher level can design smaller default boxes. However, the higher levels of the general feature pyramids are invalid for small objects. This designed structure ensures the invariance of the resolution of the feature maps and avoids the problem that the low-resolution feature map limits the design of the minimum default box. Furthermore, the default box corresponding to the feature map can effectively cover the detection area at each scale, and larger detection targets also have denser default boxes. The size and aspect ratio of the default boxes corresponding to the feature maps are determined by the bounding boxes of clustering our navel orange dataset.

For the backbone, it consists of three parts: a stem, a dense block and a transition layer. The dense block is similar to FSN and is designed by dense connectivity. The transition layer connects the dense block and the feature skyscraper network. We add an Squeeze-and-Excitation (SE) layer [5] to both the stem and transition modules. And all activation functions in detecting features. The size and aspect ratio of the default boxes corresponding to the feature maps are determined by the bounding boxes of clustering our navel orange dataset.

For the backbone, it consists of three parts: a stem, a dense block and a transition layer. The dense block is similar to FSN and is designed by dense connectivity. The transition layer connects the dense block and the feature skyscraper network. We add an Squeeze-and-Excitation (SE) layer [5] to both the stem and transition modules. And all activation functions in detecting features. The size and aspect ratio of the default boxes corresponding to the feature maps are determined by the bounding boxes of clustering our navel orange dataset.
the architecture use the Swish [6], except that SE layer uses a sigmoid function.

On the one hand, as shown in Fig. 1, we compare performance of our model with some state-of-the-art and classical one-stage models, such as SSD [7], DSOD [3], YOLOv2 [8], YOLOv3 [9], RFB [10] and FSSD [11]. And we also experiment with the performance of these models under different backbones, such as DarkNet19 [8], DarkNet53 [9], MobileNetV1 [12], MobileNetV2 [13], ResNet50 [14] and VGG16 [15]. The experimental results confirm that the FSD is the most compact architecture for the best performance in real time. And the feature skyscraper network designed and trained based on dataset statistics have significant advantages. On the other hand, our model has high accuracy for navel orange classification, compared to other orange defect detection methods. Finally, we also evaluate the effectiveness of adding the SE layer and using the Swish activation function for the backbone, and the effectiveness of the design for FSN. The main contributions in this paper:

1. We build a navel orange dataset to detect the stem end and the blossom end from the black spot.
2. We propose a feature skyscraper detector (FSD), which is optimized based on the statistical results of the dataset.
3. We design the feature skyscraper network which is better than feature pyramids, under our dataset.
4. We show that the FSD can implement state-of-the-art performance on our navel orange dataset and verify the effectiveness of the optimization for its architecture.

This paper first introduces the detection of surface defect in navel oranges and some of the current detection techniques in Sect. 2. Then, Sect. 3 shows the architecture we proposed and the settings for its training. In Sect. 4, the basics of the image acquisition device, the navel orange dataset and the statistical results for it are shown. Next, we present the details of the model implementation, compare it with the state-of-the-art deep-learning-based models and other orange defect detection methods. Then, in Sect. 5 verifies the validity of the design and improvement in the model. The final Sect. 6 concludes this paper.

2 Related work

Before the deep learning explosion, the orange peel defects detection task is generally implemented by machine learning. Behera et al. [16] classify the orange disease and compute its severity by SVM with K-means clustering and Fuzzy logic. Rong et al. [17] propose an adaptive lightness correction algorithm to solve the problem that the uneven distribution of lightness on the surface of the navel orange is difficult to detect in the dark region. However, the performance and generalization of these methods are not good, and the design of the model depends on the environment. Zhang et al. [18] identify the apple’s bruises and blemishes from the stem end and calyx of apple images by near-infrared spectrum light, but its hardware cost is relatively high. The study by Kamilaris et al. [2] shows that deep learning algorithms are indeed superior in accuracy to existing used image processing techniques. So, we consider the current detection architecture and some techniques in deep learning to design our model based on statistics of dataset.

Detection Methods Since the deep learning boom, there are mainly two types of detectors, the two-stage detection framework and the one-stage detection framework [19,20]. Among them, because it is a multistage complex pipeline, the training process of the two-stage detection framework (Representative RCNN [21], Fast RCNN [22], Faster RCNN [23] and RFCN [24] etc.) is more complicated, the optimization is difficult, and the time of inference is very slow [19]. In comparison, one-stage detection represented by YOLO [25], and SSD [7], etc., although achieving relatively low object detection quality, can avoid the problems mentioned above.

YOLO [25] puts the detection problem as a regression problem and directly outputs the relevant information of the entire image. SSD [7] applies multi-scale feature maps for detection at multi-scales and uses small convolutional filters applied to feature maps to get information about category and location. Backbones It is critical for backbones in the object detection task [26]. Representative backbones include AlexNet [27], VGG [15], Inception series [28–30], ResNet [14], SENet [5], DenseNet [4], MobileNet series [12,13], and so on. Although many backbones are designed for classification, as the classification performance increases, the performance of the object detection is also improved [26,31]. Due to the superiority of dense connectivity, we mainly consider the DenseNet.

Fusion of layers It is typical for feature pyramid to integrate information from different feature maps [20]. Yi Lin et al. [32] propose the feature pyramid network to fuse the feature maps. And Zuoxin et al. [11] design a structure in FSSD to make it easier to fuse the feature maps from different scales. M2Det [33] introduces more complex multi-level multi-scale features to detect complex small object. But for simple detection tasks, these pyramid feature fuses make the model more complicated and training difficult. Instead, we apply dense connectivity, and it is introduced in DSOD [3], which makes it inherit the features of DenseNet [4] about training from scratch. The connection method is derived from ResNet [14], but it helps to better propagate features and losses and reduce the number of model parameters [4].

Channel-wise Attention Mechanism Squeeze and Excitation [5] is one of channel attention mechanism, which can improve performance by a flexible way and only need a few additional computational cost [19]. Applying it, both CliqueNet [34] and M2Det [33] get a performance boost.
Activation Functions Ramachandran et al. [6] discover a novel activation function named Swish by automatic search techniques. And compared to the usual activation functions such as ReLU [35], PReLU [36], ELU [37], and SELU [38] etc., they confirm that Swish has the best empirical performance. This is further confirmed under our model FSD.

3 The proposed model

In this section, we first introduce the architecture of the FSD and give the specific details of the FSN and dense multi-scale features. Next, we show the design of stem and transition layer in the backbone. Finally, give the network configurations for the training.

3.1 Model architecture

We present the structure of FSD in Fig. 3, which consists of three parts: a backbone, FSN and detection layers. The backbone for feature extraction is mainly composed of a stem, a dense block and a transition layer in series. To make the model train from scratch, we use dense connectivity between each of the shaft layers in the dense block and FSN. And the architecture of the detection layer is achieved using the method by [7]. Then, all activation functions in FSD are Swish [6] which is defined as $f(x) = x \cdot \text{sigmoid}(\beta x)$, except that SE layer uses a sigmoid function.

3.2 Feature skyscraper network

The feature skyscraper network (FSN) is composed of a series of shaft layers connected by dense connectivity. The shaft layer has a convolutional layer of $1 \times 1$ and $3 \times 3$, and there is a batch normalization (BN) layer and a Swish activation layer before each convolution. As shown in Fig. 3, each input and output of the shaft layer is concatenated together as input to the next shaft layer. After concatenated, the feature map for each layer is sent to the detection layers.

The resolution $r \times r$ of the feature map depends on the image resolution $R \times R$ of the input model and the minimum side length ($W_b$ or $H_b$) of the bounding box. So, there is:

$$r = \left\lfloor \frac{R}{\min(W_b, H_b)} \right\rfloor.$$

To alleviate the channel accumulation caused by feature reuse, we design a dynamic growth rate $k$. The growth rate $k$ of the $i$th shaft layer is $(L - i + 1) \times C$, where $L$ represents the total number of shaft layers, and $C$ represents the number of
Table 1  SE layer improves model performance

|                | FSD       |
|----------------|-----------|
| Stem (SE Layer)| ✓ ✓ ☒ ✓ ✓|
| Transition (SE Layer) | ✓ ☒ ✓ ✓ |
| mAP            | 81.57 83.61 87.11 87.48 |

See Table 9 for more details

output channels of the $L$th shaft layer (Before concatenated, it is also the k of the $L$th shaft layer).

### 3.3 Dense multi-scale features

Different from the general feature pyramids, the most remarkable feature of our design FSN is that the resolution of feature maps is constant. However, this does not mean losing the characteristics of multi-scale features. In fact, our multi-scale characteristics are implemented by default boxes of different scales for each layer. This means that larger scale feature maps will be more denser.

To adapt to the detection target with a large range of size changes, one-stage models such as SSD and DSOD adopt six standard scale (0.2 0.9) default boxes. The scale (0.267) of our largest detection object is comparable to the SSD minimum default box ratio corresponding to the feature map that completely covers the original image. And the upper bound is $\max(W_b, H_b)/R$. Within this range, we use k-means clustering to determine the scale of the default box for each layer.

### 3.4 The backbone of FSD

The dense block in the backbone has the same structure as the feature skyscraper, but the growth rate $k$ is fixed. We add Channel-wise Attention Mechanism in FSD’s transition layer and stem block, as shown in Fig. 4. Many models [33,34,39,40] have introduced this method and it can effectively improve the performance. Our SE layer consists of an average pooling layer, two fully connection layers, one Swish activation function layer and one sigmoid activation function layer. And it is added between the convolution layer and the

Table 2  FSD architecture configuration

| Module                                | Output size input $3 \times 150 \times 150$ | FSD       |
|---------------------------------------|---------------------------------------------|-----------|
| Stem                                  |                                             |           |
| Convolution layer                     | $48 \times 150 \times 150$                  | $3 \times 3$, 48 conv, stride 1, padding 1 |
| Convolution layer                     | $96 \times 150 \times 150$                  | $3 \times 3$, 96 conv, stride 1, padding 1 |
| SE layer                              | $96 \times 150 \times 150$                  | Global avg pool + [96, 32, 96]fc |
| Pooling layer                         | $96 \times 75 \times 75$                    | $2 \times 2$ max pool, stride 2 |
| Dense block                           |                                             |           |
| Shaft layer 1                         | $144 \times 75 \times 75$                   |           |
| Shaft layer 2                         | $192 \times 75 \times 75$                   |           |
| Shaft layer 3                         | $240 \times 75 \times 75$                   |           |
| Shaft layer 4                         | $288 \times 75 \times 75$                   |           |
| Shaft layer 5                         | $336 \times 75 \times 75$                   |           |
| Transition layer                      |                                             |           |
| Convolution layer                     | $336 \times 75 \times 75$                   | $1 \times 1$, 336 conv, stride 1 |
| SE layer                              | $336 \times 75 \times 75$                   | Global avg pool + [336, 168, 336]fc |
| Pooling layer                         | $336 \times 38 \times 38$                   | $2 \times 2$ max pool, stride 2 |
| Feature skyscraper network (FSN)      |                                             |           |
| Shaft layer 1                         | $480 \times 38 \times 38$                   | $1 \times 1$, 480 conv $3 \times 3$, 480 conv, stride 1, padding 1 |
| Shaft layer 2                         | $576 \times 38 \times 38$                   | $1 \times 1$, 576 conv $3 \times 3$, 576 conv, stride 1, padding 1 |
| Shaft layer 3                         | $624 \times 38 \times 38$                   | $1 \times 1$, 624 conv $3 \times 3$, 624 conv, stride 1, padding 1 |
| Detection layers                      | –                                           | –         |

The growth rate for dense block is $k = 48$, and for FSN is $k = 144, 96, 48$. Each shaft layer corresponds the sequence BN-Swish-Conv and each convolution layer is the sequence Conv-BN-Swish.
### Table 3 Bounding boxes statistics

|        | W (pixel) | H (pixel) | Aspect | Area (pixel²) |
|--------|-----------|-----------|--------|---------------|
| Mean   | 12.13     | 8.389     | 1.483  | 113.095       |
| Std    | 5.307     | 3.42      | 0.478  | 88.785        |
| Min    | 4.054     | 4.008     | 0.494  | 16.858        |
| Max    | 40.057    | 31.797    | 3.996  | 724.138       |

Fig. 5 The high-performance orange grading machine

max pooling layer. In the subsequent experiments, it is also confirmed that the addition of the structure also improved the detection accuracy of our model, as shown in Table 1.

### 3.5 Network configurations

The implementation details of FSD are shown in Table 2. The resolution of each feature map of the FSN output is 38 × 38, and the growth rate $k$ of the 3 shaft layers are: 144, 96 and 48. As shown in Table 3, the average aspect ratios of the bounding boxes in dataset are 1.483, and we set it as the aspect ratios of the default boxes. Then we use K-means to calculate the three cluster centers (They are: 59.75, 165.8, and 349.0.) of the area size as the scale of the default boxes.

### 4 Experiment

In this section, first describe how we acquire the images by the acquisition device, as well as some of its parameters and task requirements. Then introduce the construction of the dataset and the statistical characteristics of the dataset. Next, we conduct and evaluate all experiments on our orange dataset benchmark, and present model performance measured by mean Average Precision (mAP), parameters and frames per second (FPS). Finally, the implementation details of all experiments and the comparisons with other approaches are performed.

#### 4.1 Image acquisition

The experimental RGB color images are collected at resolution 1280 × 1024 from a navel orange grading machine, as shown in Fig. 5. The machine vision part is mainly composed of high-resolution industrial cameras above the conveyor belt with rollers, LED warm light sources for providing sufficient light to the camera, and photoelectric switch for controlling image capture.

The navel orange triggers the photoelectric switch to capture images by the camera which rate up to 16 frames per second. At the same time, with the roller to drive the navel orange rotation, the camera can capture a series of images at different angles of an orange. After passing through the machine vision device, the navel oranges on the orbit are popped out by the spring device to classify according to the detection results of the machine vision part.

The machine vision part can obtain 11 different angles images of each navel orange to ensure that sufficient surface information is provided. And after preprocessing these images, the orange region of each image is extracted. As shown in Fig. 6, they are the same height and different width 11 images. And in order to meet the requirement of system real-time detection, the time to get result from the detection algorithm is less than 9 ms per image (111 FPS).

#### 4.2 Dataset

We collected 11,187 images from the machine vision part. Each image is required to be marked with the stem end, the blossom end, and the black spot in Pascal VOC format. In order to balance every detection object of dataset, we extracted the dataset based on the minimal mark of category.

There are 3482 color images of navel oranges labeled with 1583 stem ends, 1482 blossom ends and 2250 black spots as experimental dataset. Each detection object is marked with a bounding box as shown in Fig. 7. One tenth of these images

---

\(^1\) This machine is jointly developed by Jiangxi Reemoon Sorting Equipment Co., Ltd. and Institute of Microelectronics of the Chinese Academy of Sciences.
The red one annotated the stem end.

The blue one annotated the blossom end.

Fig. 7 The detection object is annotated with a bounding box. Those yellow bounding boxes are black spots

| Table 4 The navel orange dataset |
|----------------------------------|
|                                | Train | Test | Total |
|---------------------------------|-------|------|-------|
| Stem end                        | 1418  | 165  | 1583  |
| Blossom end                     | 1335  | 147  | 1482  |
| Black spot                      | 2012  | 238  | 2250  |
| Total                           | 4765  | 550  | 5315  |
| Images                          | 3134  | 348  | 3482  |

(348) are randomly selected as the test set, and the remaining images (3134) are used as training sets. Table 4 gives more details about this dataset.

### 4.3 Statistics

Since the input to our architecture is 150 × 150, the images with bounding boxes are resized to 150 × 150 resolution before statistics. Then we get the width and height information of all bounding boxes in the dataset. As shown in Table 3, the width-height aspect ratio and area size are then calculated, because the architecture of our model and the selection of the default boxes are related to these two statistics in subsequent experiments. Next, further visualize these information and results, as shown in Fig. 8.

**Area size** As shown in Fig. 8, the area of bounding boxes is generally small, which is 0.02672–0.267 (Calculated by min(W, H)/150 and max(W, H)/150). And according to the Table 3 results, the maximum area size is only 3.218% of the full image. This implies that the selection of the aspect ratios of the default boxes only needs to consider a few feature values.

**Aspect ratios** It is worth noting that the aspect ratio is not symmetrical distribution in 1 nearby and its distribution is also relatively concentrated as shown in Fig. 8. This implies that the selection of the aspect ratios of the default boxes only needs to consider a few feature values.

### 4.4 Implementation details

We implement the FSD based on Pytorch framework and train them from scratch using SGD with initial learning rate 0.01, 0.9 momentum, and 0.0001 weight decay on Nvidia TitanX cuDNN v6.0.21 with Intel Xeon E5-2683 v3 @2.00GHz. Then Stochastic Gradient Descent with Restarts (SGDR) makes the learning rate gradually decrease through training, and other training strategies follow SSD. For each scale feature map of the output, we use the same L2 normalization technique as DSOD do. In training, our model inherits the training methods of DSOD and the data augmentation of SSD.

### 4.5 Comparison with state-of-the-art

We uniformly set the batch size to 8, and the training epoch to 300, which is compared with other state-of-the-art models under our navel orange dataset. In addition to the DSOD and FSD training from scratch, other comparison models are loaded with the best weights of VOC pre-training for training and the default box scale and size are set by default. The training code and detailed parameter settings of the comparison model are available.

Intuitively, as shown in Fig. 1, the FSD achieves the highest mAP with the smallest number of parameters, and the inference speed fully meets the real-time requirements. More specifically, as shown in Table 5, FSD has an absolute advantage over most models. It is worth noting that FSD exceeds 1.021% of YOLOv3-Darknet53 with the highest mAP 87.479%, and its parameters is only 2.47% of YOLOv3-Darknet53. And FSD only increases the 15.181% of mAP with 11.76% of the DSOD parameters. Figure 9 shows the inference results of FSD. It is worth noting that YOLOv3 and FSSD use the FPN-style feature pyramid, and the RFB uses the SSD-style feature pyramid. YOLOv2 changes the input image and network size to achieve multi-scale, and similar strategy is also applied to [41].

### 4.6 Comparison with other orange defect detection methods

Based on statistics for the prediction boxes output by the FSD, we classify navel oranges and compare the results with other orange defect detection methods. For example, if FSD detects that an image contains a stem end and at least one black spot, then the image is classified as “Stem End” and “Black Spot”. The main processes of

---

2 https://github.com/ShuangXieIrene/ssds.pytorch.
Fig. 8 Histograms of the distribution of the area size and aspect ratio of the bounding boxes

### Table 5 Comparison of FSD with some state-of-the-art models

| Model      | Backbone       | Pre-train | SPEED (FPS) | #Parameters (MB) | mAP | Stem end | Blossom end | Black spot | Input |
|------------|----------------|-----------|-------------|------------------|-----|----------|-------------|------------|-------|
| SSD        | Darknet19      | ✓         | 103.038     | 89.419           | 66.837 | 92.364 | 75.051 | 33.097     | 300   |
| SSD        | Darknet53      | ✓         | 137.824     | 168.753          | 70.413 | 92.78   | 79.944 | 38.516     | 300   |
| SSD        | MobilenetV1    | ✓         | 50.966      | 21.106           | 60.215 | 90.14   | 66.884 | 23.62   | 300   |
| SSD        | MobilenetV2    | ✓         | 94.367      | 12.624           | 66.689 | 93.457 | 72.626 | 33.983 | 300   |
| SSD        | Resnet50       | ✓         | 99.237      | 46.494           | 68.06  | 92.148  | 75.783 | 36.25 | 300   |
| SSD        | VGG16          | ✓         | 95.341      | 91.896           | 69.569 | 89.418  | 77.708 | 41.579 | 300   |
| RFB        | Darknet19      | ✓         | 163.49      | 118.798          | 69.757 | 91.287 | 78.239 | 39.745 | 300   |
| RFB        | Darknet53      | ✓         | 118.975     | 198.132          | 74.115 | 95.397 | 79.96 | 46.988 | 300   |
| RFB        | MobilenetV1    | ✓         | 164.133     | 33.446           | 66.621 | 91.964 | 74.172 | 33.728 | 300   |
| RFB        | MobilenetV2    | ✓         | 157.72      | 14.142           | 69.179 | 91.438 | 77.566 | 38.532 | 300   |
| RFB        | Resnet50       | ✓         | 105.675     | 75.872           | 72.106 | 94.927 | 77.694 | 43.697 | 300   |
| RFB        | VGG16          | ✓         | 77.897      | 121.274          | 74.245 | 94.908 | 80.097 | 47.732 | 300   |
| FSSD       | Darknet19      | ✓         | 105.265     | 129.558          | 76.731 | 95.8   | 88.474 | 45.917 | 300   |
| FSSD       | Darknet53      | ✓         | 94.267      | 208.892          | 78.41  | 95.088 | 91.883 | 48.261 | 300   |
| FSSD       | MobilenetV1    | ✓         | 118.831     | 66.231           | 77.931 | 94.204 | 85.145 | 54.445 | 300   |
| FSSD       | MobilenetV2    | ✓         | 134.62      | 22.002           | 77.568 | 96.977 | 83.941 | 51.786 | 300   |
| FSSD       | VGG16          | ✓         | 77.073      | 122.524          | 73.16  | 96.087 | 86.201 | 37.193 | 300   |
| YOLO v2    | Darknet19      | ✓         | 150.25      | 193.002          | 80.509 | 94.274 | 86.663 | 60.59 | 416   |
| YOLO v2    | MobilenetV1    | ✓         | 148.16      | 129.674          | 80.175 | 94.911 | 86.574 | 59.04 | 416   |
| YOLO v2    | MobilenetV2    | ✓         | 127.692     | 99.563           | 58.668 | 78.597 | 54.166 | 43.239 | 416   |
| YOLO v3    | Darknet53      | ✓         | 82.577      | 235.628          | 86.458 | 94.867 | 93.71 | 70.798 | 416   |
| YOLO v3    | MobilenetV1    | ✓         | 118.625     | 92.967           | 86.317 | 97.388 | 90.069 | 71.494 | 416   |
| YOLO v3    | MobilenetV2    | ✓         | 111.549     | 85.816           | 85.406 | 93.05  | 92.018 | 71.15 | 416   |
| DSOD       | DenseNet       | ✗         | 78.842      | 49.422           | 72.298 | 92.336 | 81.053 | 43.505 | 300   |
| FSD        | –              | ✗         | 131.02      | 5.812            | 87.479 | 98.235 | 86.882 | 77.318 | 150   |

Bold values represent the best value
Fig. 9  Show the detection results of FSD by visualizing prediction boxes. The stem end and the blossom end can be detected from a bunch of disturbing black spots, and even FSD can identify them at the darker edges.

Table 6  Comparison with other orange defect detection methods

| Method | Stem end | Blossom end | Black spot |
|--------|----------|-------------|------------|
| GLCM + SVM [16] | 53.161 | 57.759 | 72.989 |
| GLCM + AG + BPNN [42] | 55.460 | 59.483 | 72.701 |
| GLCM [16] + BPNN [42] | 52.586 | 50.862 | 73.851 |
| FALCA [17] + GLCM [16] + SVM [16] | 52.874 | 58.046 | 72.989 |
| FALCA [17] + GLCM [16] + BPNN [42] | 55.172 | 57.759 | 76.437 |
| SSD-MobilenetV1 | 97.701 | 94.540 | 96.839 |
| FSD | 98.563 | 95.977 | 97.701 |

The results are for navel oranges classification, while the corresponding results in other tables are for bounding boxes.

Table 7  Comparison of feature skyscraper with several feature pyramids

| Multi-scale feature | #Parameters (MB) | Speed (FPS) | mAP | Stem end | Blossom end | Black spot |
|---------------------|------------------|-------------|-----|----------|-------------|------------|
| FSN | 5.812 | 131.02 | 87.479 | 98.235 | 86.882 | 77.318 |
| DSOD | 5.637 | 125.761 | 84.957 | 92.482 | 87.513 | 74.877 |
| FPN | 7.950 | 124.371 | 84.451 | 93.873 | 84.881 | 74.598 |
| SSD | 7.450 | 130.782 | 84.109 | 95.204 | 84.657 | 72.465 |

The backbone is based on FSD.

Other navel orange defect detection methods include feature extraction and feature classification. K-means clustering algorithm for segmentation [16] can determine the scope of the defect. Gary-level co-occurrence matrix (GLCM) [16,42] can extract texture features, and SVM [16] and BPNN [42] can classify features. So, We compare with combinations of these methods. We also add a deep-learning-based network with lower mAP (in Table 5) to compare. As illustrated in Table 6, the deep-learning-based method has obvious advantages, and FSD achieves the best performance. However, other orange defect detection methods have lower accuracy for the stem end and blossom end.
5 Discussion

On the one hand, we compare the feature skyscraper with several typical feature pyramids and explore the multi-scale feature of the FSN. On the other hand, it is further validated that the effectiveness of the design for the backbone.

5.1 Feature skyscraper versus feature pyramids

To compare the feature skyscraper and feature pyramids, we sequentially replace the FSN with the DSOD-style, SSD-style and FPN-style feature pyramid on the FSD backbone. And three levels (59.7, 165.8, 349.0) are uniformly configured for multi-scale feature maps.

As shown in Table 7, FSN has good results for complex small target stem ends and is on average 4.382% higher than the other three feature pyramids. And the black spot is also an average of 3.338% higher. Overall, the mAP of the FSN is an average of 2.973% higher and its speed is slightly superior. The resolution invariance of FSN provides dense feature maps for larger-scale default boxes, so the default boxes are effectively covered in the candidate region of the detection object. As shown in the Fig. 8 visualization, most stem ends are indeed distributed near the second-level cluster center 165.8.

5.2 Multi-scale default box

The resolution of feature maps for DSOD, SSD and FPN is reduced as the scale is reduced, which constrains the minimum effective setting size of the default box. That is because low-resolution feature map corresponds to large size default boxes, which cannot effectively set smaller values for small targets. However, FSN ensures that there is more room for the default box scale setting, which is especially effective when the detection object is generally small.

As illustrated in the first part of Table 8, the ablation study demonstrates the model performance at six kinds of scales. The mAP of multi-scale default boxes is generally higher than that of a single scale. This means the design is effectiveness for FSN to model multi-scale features. In particular, setting three shaft layers is the best. And in the second part of the table, each shaft layer is set with the same size default box. This leads to a decrease in the performance of the model, and the larger-scale default box cannot effectively detect black spots. So, multi-scale default box is the key for FSN to model multi-scale features.

5.3 Model improvement

Feature extraction In the first part of the Table 9, retaining a shaft layer as a dense block can achieve a 4.124% improvement (compared to no shaft layer) in model performance.
## Table 9  Verify the FSD module and its design effectiveness

| # Shaft layers | Nonlinear activations | Stem (SE layer) | Transition (SE layer) | Speed (FPS) | #Parameters (MB) | mAP | Stem end | Blossom end | Black spot |
|----------------|-----------------------|-----------------|-----------------------|-------------|-----------------|-----|---------|-------------|-----------|
| 0              | Swish                 | ✓               | ✓                     | 157.958     | 3.176           | 80.658 | 94.303  | 78.427      | 69.243     |
| 1              | Swish                 | ✓               | ✓                     | 143.088     | 3.632           | 84.782 | 94.398  | 86.845      | 73.104     |
| 2              | Swish                 | ✓               | ✓                     | 146.399     | 4.123           | 83.774 | 91.686  | 84.009      | 75.626     |
| 3              | Swish                 | ✓               | ✓                     | 140.976     | 4.65            | 85.799 | 94.132  | 90.507      | 72.757     |
| 4              | Swish                 | ✓               | ✓                     | 127.848     | 5.213           | 86.256 | 95.705  | 90.544      | 72.519     |
| 5              | Swish                 | ✓               | ✓                     | 131.02      | 5.812           | 87.479 | 98.235  | 86.882      | 77.318     |
| 6              | Swish                 | ✓               | ✓                     | 126.276     | 6.449           | 85.691 | 96.158  | 85.389      | 75.525     |
| 7              | Swish                 | ✓               | ✓                     | 119.525     | 7.12            | 84.597 | 94.28   | 87.66       | 71.852     |
| 5              | elu                    | ✓               | ✓                     | 129.221     | 5.812           | 84.846 | 95.252  | 85.644      | 73.644     |
| 5              | leaky relu             | ✓               | ✓                     | 135.198     | 5.812           | 84.804 | 91.176  | 88.362      | 74.874     |
| 5              | prelu                  | ✓               | ✓                     | 131.586     | 5.812           | 85.888 | 96.714  | 86.494      | 74.454     |
| 5              | relu                   | ✓               | ✓                     | 128.031     | 5.812           | 85.318 | 93.067  | 88.787      | 74.101     |
| 5              | selu                   | ✓               | ✓                     | 132.222     | 5.814           | 81.415 | 93.408  | 78.705      | 72.133     |
| 5              | sigmoid                | ✓               | ✓                     | 126.218     | 5.812           | 81.836 | 90.605  | 82.758      | 72.146     |
| 5              | tanh                   | ✓               | ✓                     | 134.4       | 5.812           | 83.241 | 93.921  | 83.551      | 72.25      |
| 5              | Swish                  | ✓               | ✓                     | 135.267     | 5.787           | 87.111 | 97.528  | 88.287      | 75.548     |
| 5              | Swish                  | ✓               | ✓                     | 129.689     | 5.522           | 83.61  | 93.7    | 87.505      | 69.624     |
| 5              | Swish                  | ✓               | ✓                     | 133.153     | 5.498           | 82.171 | 91.289  | 83.427      | 71.795     |

The models in the table are all modified based on the FSD, and the input resolution is uniformly set to 150. Where #Shaft Layers are in the backbone part

Bold values represent the FSD
So, we confirm that it is necessary to design at least one shaft layer as a dense block, and the shaft layer is set to 5 layers to make the mAP achieve better results and have a reasonable #Parameters.

**Activation functions** In the second part of the Table 9, we replaced the activation function in the architecture and found that the effect of the Swish activation function is optimal. By replacing the commonly used ReLU activation function with Swish, the overall performance of the model is improved by 2.161%.

**Attention mechanism** In the third part of the Table 9, we verified the validity of adding the SE layer. It is proved in the experiment that adding the SE layer to the stem and the transition layer, respectively, not only improves the model, but both adding them also makes the model further improved. We visualized the feature map before inputting the SE layer and after outputting the SE layer, in Fig. 10.

**6 Conclusion**

In this work, to accurately and efficiently distinguish the stem end and the blossom end of navel orange from its black spot, we built a navel orange dataset and proposed the feature skyscraper detector (FSD) model based on the dataset statistical properties. Different from the general feature pyramids, the feature skyscraper network (FSN), the main part of the model, was proposed for multi-scale small objects, and combined with the design of multi-scale default boxes to generate dense multi-scale feature maps. Adding the attention mechanism and application of dense connectivity and Swish in the backbone further improved the performance of the model. In the experiment, FSD achieved mAP of 87.479% at 131 FPS with only 5.812M parameters, and which mAP exceeded 15.181% of the DSOD that also applied dense connectivity. It is competitive with the state-of-the-art one-stage detectors like SSD, DSOD, YOLOv2, YOLOv3, RFB and FSSD with high detection accuracy, most compact architecture and real-time performance. Compared with other orange defect detection methods, FSD showed the best results for orange classification. And we verified that FSN is better than the general feature pyramids under our dataset. In particular, the average precision of the stem end reached 98.235%, the blossom end reached 86.882%, and the black spot reached 77.318%.

**Acknowledgements** This work was supported by National Key R&D Program of China (No. 2018YFD0700300).

**References**

1. Korf, H.J.: Survival of *Phylllosticta citricarpa*, Anamorph of the Citrus Black Spot Pathogen. University of Pretoria, Pretoria (1998)
2. Kamarilas, A., Prenafeta-Boldu, F.: Deep learning in agriculture: a survey. Comput. Electron. Agric. **147**, 70–90 (2018). https://doi.org/10.1016/j.compag.2018.02.016
3. Shen, Z., Liu, Z., Li, J., Jiang, Y.-G., Chen, Y., Xue, X.: Dsod: learning deeply supervised object detectors from scratch. In: Proceedings of the IEEE International Conference on Computer Vision 2017, pp. 1919–1927
4. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Ieee: densely connected convolutional networks. In: 30th Ieee/Cvf Conference on Computer Vision and Pattern Recognition (Cvpr), Honolulu, HI (2017)
5. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. ArXiv e-prints (2017)
6. Ramachandran, P., Zoph, B., Le, Q.V.: Searching for activation functions. arXiv:1710.05941 (2017)
7. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: SSD: Single shot multibox detector. In: European Conference on Computer Vision 2016, pp. 21–37. Springer
8. Redmon, J., Farhadi, A.: Yolo9000: better, faster, stronger. arXiv preprint (2017)
9. Redmon, J., Farhadi, A.: Yolov3: an incremental improvement. arXiv:1804.02767 (2018)
10. Liu, S., Huang, D.: Receptive field block net for accurate and fast object detection. In: Proceedings of the European Conference on Computer Vision (Eccv) 2018, pp. 385–400
11. Li, Z., Zhou, F.: Fssd: feature fusion single shot multibox detector. arXiv:1712.00960 (2017)
12. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: efficient convolutional neural networks for mobile vision applications. arXiv:1704.04861 (2017)
13. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.-C.: Mobilenetv2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2018, pp. 4510–4520
14. He, K., Zhang, X., Ren, S., Sun, J.: Ieee: deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (Cvpr), p. 2016. Seattle, WA (2016)
15. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556 (2014)
16. Behera, S.K., Jena, L., Rath, A.K., Sethy, P.K.: Disease classification and grading of orange using machine learning and fuzzy logic. In: 2018 International Conference on Communication and Signal Processing (Iccsp) 2018, pp. 0678–0682. Ieee (2018)
17. Rong, D., Ying, Y., Rao, X.: Embedded vision detection of defective orange by fast adaptive lightness correction algorithm. Comput. Electron. Agric. 138, 48–59 (2017)
18. Zhang, D., Lillywhite, K.D., Lee, D.-J., Tippett, B.J.: Automated apple stem end and calyx detection using evolution-constructed features. J. Food Eng. 119(3), 411–418 (2013)
19. Liu, L., Ouyang, W., Wang, X., Fieguth, P., Chen, J., Liu, X., Pietikainen, M.: Deep learning for generic object detection: a survey. arXiv:1809.02165 (2018)
20. Agarwal, S., Terrail, J.O.D., Jurie, F.: Recent advances in object detection in the age of deep convolutional neural networks. arXiv:1809.03193 (2018)
21. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2014, pp. 580–587
22. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision 2015, pp. 1440–1448
23. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems 2015, pp. 91–99
24. Dai, J., Li, Y., He, K., Sun, J.: R-fcn: object detection via region-based fully convolutional networks. In: Advances in Neural Information Processing Systems 2016, pp. 379–387
25. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2016, pp. 779–788
26. Huang, J., Rathod, V., Sun, C., Zhu, M., Korattikara, A., Fathi, A., Fischer, I., Wojna, Z., Song, Y., Guadarrama, S.: Speed/accuracy trade-offs for modern convolutional object detectors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2017, pp. 7310–7311
27. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. Commun. ACM 60(6), 84–90 (2017). https://doi.org/10.1145/3065386
28. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. ArXiv e-prints (2015)
29. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. ArXiv e-prints (2015)
30. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.: Inception-v4, Inception-ResNet and the impact of residual connections on learning. ArXiv e-prints (2016)
31. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. 115(3), 211–252 (2015)
32. Lin, L., Dollr, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2017, pp. 2117–2125
33. Zhao, Q., Sheng, T., Wang, Y., Tang, Z., Chen, Y., Cai, L., Ling, H.: M2Det: a single-shot object detector based on multi-level feature pyramid network. arXiv:1811.04533 (2018)
34. Yang, Y., Zhong, Z., Shen, T., Lin, Z.: Convolutional neural networks with alternately updated clique. ArXiv e-prints (2018)
35. Girotos, X., Bordes, A., Bengio, Y.: Deep sparse rectifier neural networks. In: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics 2011, pp. 315–323
36. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: surpassing human-level performance on imagenet classification. In: Proceedings of the IEEE International Conference on Computer Vision 2015, pp. 1026–1034
37. Clevert, D.-A., Unterthiner, T., Hochreiter, S.: Fast and accurate deep network learning by exponential linear units (elus). arXiv:1511.07289 (2015)
38. Klambauer, G., Unterthiner, T., Mayr, A., Hochreiter, S.: Self-normalizing neural networks. In: Advances in Neural Information Processing Systems 2017, pp. 971–980
39. Chen, L., Zhang, H., Xiao, J., Nie, L., Shao, J., Liu, W., Chua, T.-S.: Sca-cnn: spatial and channel-wise attention in convolutional networks for image captioning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition 2017, pp. 5659–5667
40. Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., Wang, X., Tang, X.: IEEE: residual attention network for image classification. In: 30th IEEE Conference on Computer Vision and Pattern Recognition. IEEE Conference on Computer Vision and Pattern Recognition, pp. 6450–6458 (2017)
41. Zeng, Y., van der Lubbe, J., Loog, M.: Multi-scale convolutional neural network for pixel-wise reconstruction of Van Goghs drawings. Mach. Vis. Appl. 30(7–8), 1229–1241 (2019)
42. Thendral, R., Suhasini, A.: Genetic algorithm based feature selection for detection of surface defects on oranges. J. Sci. Ind. Res. 75(9), 540–546 (2016)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.