Fruit Detection from Digital Images Using CenterNet

Kun Zhao(✉) and Wei Qi Yan(✉)

Auckland University of Technology, Auckland 1010, New Zealand
{kvz5449,weiqi.yan}@aut.ac.nz

Abstract. In this paper, CenterNet is chosen as the model to settle fruit detection problem from digital images. Three CenterNet models with various backbones were implemented, namely, ResNet-18, DLA-34, and Hourglass. A fruit dataset with four classes and 1,690 images was established for this research project. By comparing those models, followed the experimental results, the deep learning-based model with DLA-34 was selected as the final model to detect fruits from digital images, the performance is excellent. In this paper, the contribution is that we deploy a model based on CenterNet for visual object detection to resolve the problem of fruit detection. Meanwhile, there are 1,690 images distributed in four classes. Throughout evaluating the performance of the model, we eventually affirm the CenterNet based on DLA-34 to detect multiclass fruits from our images. The performance of this method is better than the existing ones in fruit detection.

Keywords: CenterNet · ResNet-18 · DLA-34 · Hourglass net · Fruit detection

1 Introduction

Fruit industry, as a typical one with high economic value, has an intensive requirement for automation [1]. In the fruit industry, the cost spent on picking holds the dominant percentage of the whole cost. A large amount of electric power, fuel, irrigation, and chemical fertilizer are demanded in agriculture development. The speed, cost, and safety of picking directly affect the final output and quality of fruit production. Hence, more and more harvesting robots are being deployed in the fruit industry to reduce the cost of picking and improve the quality of fruit [1]. For harvesting robots, a rich assortment of tasks needs to be handled, such as detection, picking, localization, classification, selection, and grading.

Among these missions, visual object detection is the most critical one [33, 34], hence, we should settle this problem firstly [2]. The follows show how to detect visual objects from an image, which is thought as an answer to object detection in computer vision [2]. The methods to handle this research problems mainly have been categorized into two groups: Machine learning-based (ML-based) method and deep learning-based (DL-based) method [3, 31, 32].

Owing to those methods, in this paper, we expect to design a new model to resolve this fruit detection problem. What the ML-based methods as the base of visual object

M. Nguyen et al. (Eds.): ISGV 2021, CCIS 1386, pp. 313–326, 2021.
https://doi.org/10.1007/978-3-030-72073-5_24

detection are the main process of how to deploy a model for object detection. However, those methods are not working well as they used to be due to the development of the theory and hardware in the past decades. Therefore, we design a DL-based model that could settle fruit detection fast and accurately.

The main purpose of this project is to find a practical model that could resolve this problem with object detection for fruits. Therefore, we will analyze those existing ML-based methods. Consequently, we detail the advantages and disadvantages. Finally, we choose the most appropriate method for this project and create a new model. Furthermore, we need to test and evaluate this model and confirm that this model is robust and accurate by using our own dataset. At last, CenterNet is treated as an effective method for this paper. Moreover, it will be tested based on three backbone models and the most suitable one will be picked up. In this paper, we also collect a new dataset for four classes of fruits. Besides, we train and test the model by using our dataset.

The contributions of this paper are: (1) We design a model to handle fruit detection problems based on CenterNet; (2) A dataset with four classes of fruits for this research project has been created; (3) Training and testing this model by using our dataset, as well as, evaluating the performance of this model are delineated in this paper.

The remaining parts of this paper are organized as follows. We review literature in Sect. 2, our method is presented in Sect. 3, the results are showcased in Sect. 4, our conclusion and future work will be addressed in Sect. 5.

2 Literature Review

A vast majority of research work has been conducted in visual object detection. Many methods have been designed and implemented. The methods are grouped into twofold depending on a theoretical basis. The first one is the machine learning-based (ML-based) methods, the second one is the deep learning-based (DL-based) methods [4].

ML inspires us in multiple ways, especially for pattern classification from huge amounts of high dimensional images, which also brings in the development of computer vision [5]. After obtained an image, the sliding window method will be applied to generate candidate bounding boxes. This process is named as region selection, which means the region we expect to detect has been selected into these boxes. After this step, feature extraction will be commenced for each candidate bounding box. The feature inside each box will be extracted from the image by using specific methods. Finally, a classifier will be employed to classify each box based on its features [6].

In 2004, Viola and Jones put forward a method to detect visual object by using Haar feature [7], which takes use of AdaBoost as the classifier to classify each box based on already extracted Harr feature. In 2005, a method was proposed based on SVM and HOG [8]. A method DPM was put forward in 2008 [9]. Similarly, the classifier of DPM is SVM which enhances the HOG by using a combing signed gradient with an unsigned gradient to make it richer so as to express a broad spectrum of visual objects. Meanwhile, it utilizes PCA (i.e., principal component analysis) to work for dimension reduction so as to reduce complexity and accelerate the computational speed. Hence, it could get a balance between the complexity and the speed of classification.

Regarding ML-based methods, the features generally are extracted for a class of specific objects [10]. Thus, the visual features for object detection generally are various.

In another word, those features are not transportable, the designer needs to specify other features for a different object [10]. Besides, the sliding window method will generate a pretty rich number of bounding boxes to detect the visual object, which will waste considerable computational resources [11].

We know that the ML-based methods are applied to resolve the object detection problem. The conception of deep learning (DL) comes from artificial neural networks (ANNs), which essentially means a kind of specific structure with a depth of hidden layers. The ANNs were inspired by research outcomes of our human brain, which deals with various tasks by simulating the mechanism of human neurons. The first and the most basic model was named as the MP (i.e., McCulloch and Pitts) model, which is based on a class of artificial neural cells [12].

The base of current deep neural networks was regarded as perceptron in 1958 [12]. The original purpose was to handle binary classification problems. In 1970, automatic differentiation was put forward, which is based on BP (i.e., Backpropagation) algorithm, it is a crucial method for improving the speed of ANN training [13]. Combing BP and ANNs together inspired the subsequent MLP (Multilayer Perception) [14] in 1986.

Furthermore, the activation function was replaced by a sigmoid function. This nonlinear map function could enhance the performance of MLP, which could effectively tackle the nonlinear classification problem. According to the structure of our human brain, the idea is to imitate the multilayer stacked human brain. But how to train a deep network confuses researchers for a long time. The problems of vanishing gradients and exploding gradients are the reasons why deep nets could not be trained well. The attenuation of error for BP in the deep net will exponentially be increased with the growth of the network layers [15].

In 2006, deep belief net (DBN) was designed in 2006. This network solved the problem that a DNN is hard to be trained by using layer by layer pretraining [16]. Convolutional neural networks (CNNs) as a kind of ANNs, were inspired by the visual system. The original conception was enlightened by using the visual layer of cats, as named as the receptive field [17]. In 1979, neocognitron was put forward by being combined with ANNs and receptive field, which was thought as the first CNN structure [18]. In 1989, weight sharing was brought up by LeCun [19], each convolutional kernel was used to detect a particular feature and greatly decrease the parametric quantity of CNNs so as to make the complex computations to be possible.

In 1998, LeCun combines convolutional layers and downsampling layers to design a model named LeNet, AlexNet was put forward in 2012 [20]. AlexNet has considerable progress in various improvement. Firstly, it replaces sigmoid function with another activation function, ReLU function, which tackles the gradient vanishing problem and decreases the amount of computations. Another improvement of AlexNet is to decrease overfitting by using the dropout method, which could make the whole network having a better generalization ability that does not depend on local features [20]. VGGNet as the offspring network of AlexNet has great improvements to obtain a better performance in 2014 [21]. The most important change is that VGGNet takes use of multiple convolutional kernels with a size of 3×3 to replace a big size kernel. In 2014, GoogLeNet by using the inception module parallelly executes multiple size convolutional operations [22].

A DL-based method R-CNN [22] was implemented to overcome those problems of ML-based methods. The feature extraction of R-CNN takes use of CNN instead of the HOG feature. AlexNet was chosen to extract visual feature of each box automatically. Based on this, DL-based method overcomes the disadvantage of the ML-based method in the feature extraction. For the classification, R-CNN still selects SVM as its classifier. Because of its complex calculations, the speed of R-CNN is very slow. Fast R-CNN [23] conducts visual feature extraction based on the whole image, instead of abundant region proposals. This could save the time of computations as well as memory for the CNN nodes. Faster R-CNN has a better performance than Fast R-CNN by using region proposal net (RPN) to generate region proposals. But the method to generate region proposals still spends a huge amount of time. Therefore, how to overcome this bottleneck is the improvement of Faster R-CNN [24].

A one-stage method YOLO was designed to make the network very fast [25]. YOLO directly predicts the class and location of the visual object without anchor and RPN. Hence, its speed is very fast, but its accuracy has been dipped. It does not have a prior box, the detection problem is thought as a regression problem, its structure is very simple, the speed is very fast. In order to have a similar speed of YOLO and accuracy of Faster R-CNN in object detection, SSD was designed as a new model that combines both advantages of them [26]. Furthermore, it makes use of multiscale feature maps to detect the visual object with various sizes. SSD does achieve better performance based on speed and accuracy. However, the semantic information is not enough by using the feature map as shown in Fig. 2, which leads to the difficulty for detecting a small object.

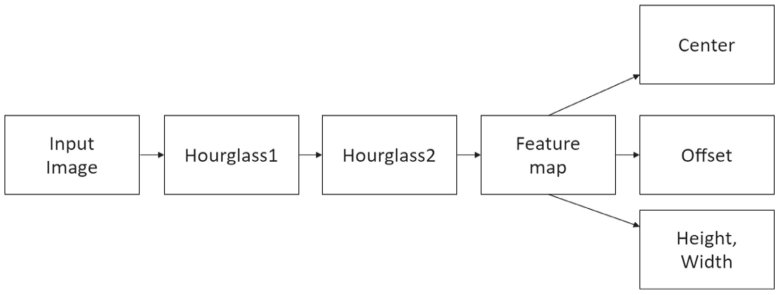

Fig. 1. The structure of CenterNet

In order to overcome the shortcoming of anchors, multiple models were designed without an anchor. Inspired by the first anchor-free model, a one-stage and anchor-free model CenterNet was proposed [27] as shown in Fig. 1. CenterNet makes use of three heads to detect an object, which conducts center pooling for the feature map to get the center. Meanwhile, it works for cascade corner pooling to obtain the offset and size. The center gets from the feature map as shown in Fig. 2, which will be transformed to a heatmap, the location of the object which has the highest value is the center. The center should be surrounded, the offset is used to assess the deviation of centers between the feature map and the original image. Hence, this model used size, center, and offset to detect the object. As an anchor-free model, it could handle with our fruit detection

problem. In this paper, we choose CenterNet as our net to accomplish the fruit detection problem from digital images.

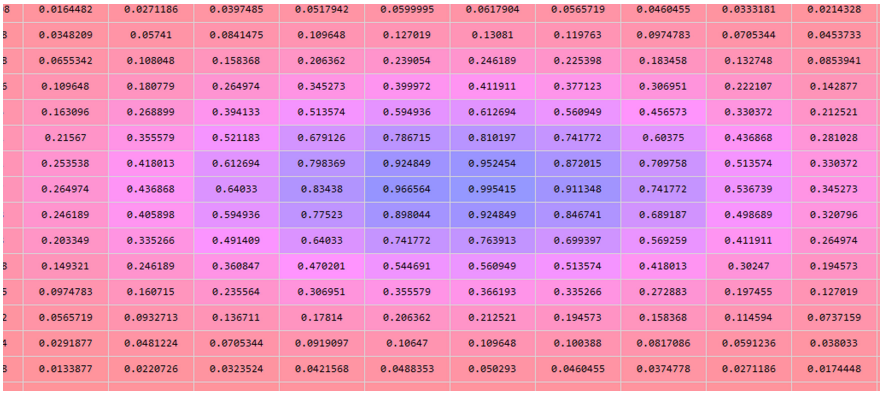

Fig. 2. A sample of heatmap

3 Our Method

The method of how CenterNet seeks the center point is that it only finds the local peak point in the heat map. Each peak point is a center of an object. Without NMS processing and anchor, it could save our running time.

Firstly, the input image was denoted as $I \in R^{W \times H \times 3}$, W is the width of our image, the height is H. The channel number of our image is 3, which means that the input image is a color one. The feature map is transformed into a keypoint heatmap through Gaussian Kernel.

For the keypoint heatmap, given $\hat{Y} \in [0, 1]^{\frac{W}{R} \times \frac{H}{R} \times C}$, R is the downsampling rate, which equals four in our project, C is the number of classes. In our dataset, we only have four kinds of fruits, hence C equals to 4. Given $\hat{Y}_{xyc} = 1$, for class c, it was detected in the (x, y) of the heatmap. On the contrary, $\hat{Y}_{xyc} = 0$ means it was not detected.

As shown in Fig. 2, the middle one is the most approximately near 1.0, that means there is an object at this location. For net training, the center point p of the ground truth needs to be calculated, $p = \left(\frac{x_1+x_2}{2}, \frac{y_1+y_2}{2}\right)$, where x and y are the coordinates in the ground truth. But the feature map was downsampled, p is also downsampled by using R, $\tilde{p} = \left[\frac{p}{R}\right]$. Thus, $\tilde{p}$ is the center point of truth data in the feature map. Then $Y_{xyc} = \exp\left(-\frac{(x-\tilde{p_x})^2+(y-\tilde{p_y})^2}{2\sigma_p^2}\right)$ as a Gaussian kernel was used to find the distribution of the keypoints in the feature map. In order to train this network, a loss function is needed to be implemented to assess its result.

$$L_{det} = L_c + \lambda_{size} L_{size} + \lambda_{off} L_{off} \tag{1}$$

where L_{det} is the total loss. In Eq. (1), the total loss contains three losses, each of which corresponds to a head of CenterNet, L_c is the loss of center, L_{size} is the loss of bounding box size, or the loss of width and height, L_{off} is the loss of offset, λ_{size} and λ_{off} are hyperparameter to modify the influence of off loss and size loss.

This model is assessed by combining these three different losses. Among them, the center loss is the most important one.

$$L_c = -\frac{1}{N}\sum_{xyz}\begin{cases}\left(1-\widehat{Y_{xyc}}\right)^a \log\left(\widehat{Y_{xyc}}\right) \; if \; Y_{xyc}=1 \\ \left(1-Y_{xyc}\right)^\beta \left(\widehat{Y_{xyc}}\right)^a \log\left(1-\widehat{Y_{xyc}}\right) \; otherwise\end{cases} \tag{2}$$

where $\alpha = 2$, $\beta = 4$. This refers to focal loss, which avoids weights having a dominant control and deals with the unbalance of the positive and negative samples. It makes use of the training process focusing on hard examples, instead of easy samples. In this way, it could tremendously decrease the weight of easy negatives.

Inspired by focal loss, the center loss is also designed to make this model focusing on useful information. The hyperparameters α, β are used to adjust the relationship between the loss of center and non-center points. Apart from the hyperparameters α, β, N is the number of keypoints of the input image, which is used to normalize the positive focal loss within the interval [0,1]. Besides, Y_{xyc} corrects the training process, $\widehat{Y_{xyc}}$ means that whether it is detected as a class or not.

During training, given $\widehat{Y_{xyc}} = 1$, $Y_{xyc} = 1$, it is a point that is easy to be detected after training. For this point, $\left(1-\widehat{Y_{xyc}}\right)^a$ will be used to minimize the L_c to let our net learn other information from different parts, to train the net better. It is a method to avoid easy samples and focus on the hard samples. In another word, it has obtained enough information for model training from these features. If trained too much, the deep learning model will be overfitting because these features will have excessive weights. Hence, this model needs to use $\left(1-\widehat{Y_{xyc}}\right)^a$ and decrease the contribution of the weights.

It is a very important idea of how this model to learn without the anchor. In order to achieve this purpose, our model will only focus on the center of each object. We need to increase the weight of the center point whilst reducing the weight of its neighbors by synthesizing $\left(1-Y_{xyc}\right)^\beta$ and $\left(\widehat{Y_{xyc}}\right)^a$. Hence, we combine the two parts of values depending on the distance between them and the center point. This will make our model learn the critical features from those points which are far from the center. In this way, it enhances the center point while undermining the neighbor points. It is a useful method to tackle the unbalance of positive and negative samples.

After trained our model by using the center loss function, the offset loss for each center point will be performed. If the feature map is remapped to the original image, it will lead to a location offset. In order to assess this offset, L_{off} will be applied.

$$L_{off} = \frac{1}{N}\sum_p \left|\widehat{O_{\tilde{p}}} - \left(\frac{p}{R}-\tilde{p}\right)\right| \tag{3}$$

After mapped the image to feature map for four times, the location of the center point will have a precision loss, which is called offset loss. To evaluate this offset loss,

we use the predicted offset $\widehat{O_{\tilde{p}}}$ to minus the truth offset $\left(\frac{p}{R} - \tilde{p}\right)$. Then, the sum of the absolute value will be averaged by using the L_1 loss method.

$$L_{size} = \frac{1}{N} \sum_{k=1}^{N} \left| \widehat{S_{p_k}} - S_k \right| \tag{4}$$

where $\widehat{S_{p_k}}$ is the size that our model predicts, $S_k = \left(x_2^{(k)} - x_1^{(k)}, y_2^{(k)} - y_1^{(k)}\right)$ is a truth size calculated by using the value after downsampling the location of the top-left and bottom-right of the dataset. Finally, the sum of all the differences will be averaged by using the L_1 norm method. As a result, the size loss will be obtained.

Thus, we see that the CenterNet combines three loss values: Center loss, size loss, and offset loss to evaluate the performance of this model. The CenterNet takes use of the center loss method, which is inspired by the focal loss to handle the hard and easy samples problem. Hence, it could make this model adaptively learn those samples and features with various weights.

The performance of CNN networks does not obtain an improvement with deepening depth. The reason is the degeneration generated by deepening the network, the SGD optimizer could not achieve a satisfactory result. ResNet was designed to overcome this problem by using shortcut structure [28]. An 18-layer residual network is chosen as the backbone of our model.

The second backbone of our model is DLA [29], which merges the feature from multiple depths. In Fig. 5, we see that DLA makes use of iterative deep aggregation (IDA) and hierarchical deep aggregation (HDA) to conduct the upsampling. The IDA part merges the feature extracted from each subnetwork grade by grade. Each red box labels that the hierarchical deep aggregation, whose block only receives the feature from the previous block. By combing IDA and HDA together, DLA is able to improve the utilization ratio of features between the layers or blocks (Fig. 3).

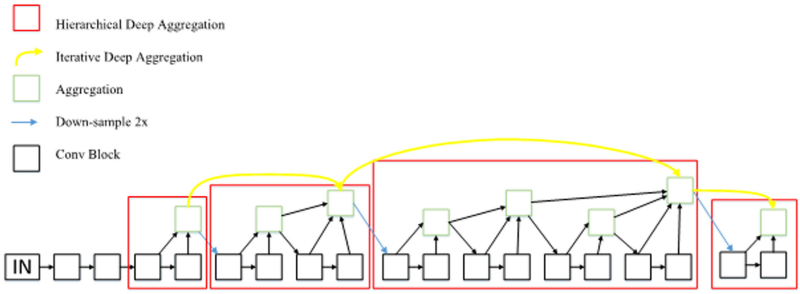

Fig. 3. The structure of DLA-34

Our last method is Hourglass network [30]. The structure is shown in Fig. 4. The original purpose of Hourglass was used to catch the information on each scale to avoid missing small local features. On the left part, from c_1 to c_4, in this bottom-up process, the image is downsampled from high resolution to low one. Meanwhile, it will extract features based on multiple scales and transport them to CNN in the up part from c_{1a} to

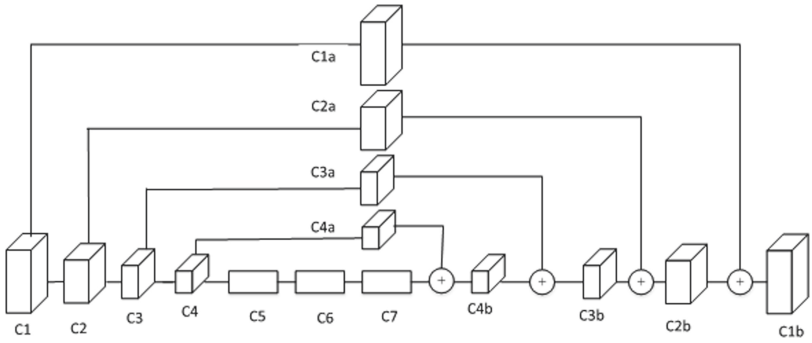

Fig. 4. The structure of Hourglass

c_{41}. On the right part, it will conduct upsampling to restore the feature map. Meanwhile, the right part will merge the feature from up and left finally to c_{1b}. Hence, Hourglass utilizes the hourglass model to extract as many features as it can. It was used as our third backbone model.

4 Our Results

4.1 Data Collection

In this paper, our task is to design a model and detect fruits from digital images. Therefore, we collect our visual data by taking photos for fruits firstly. Then, we label the fruits in each image to get the class and location of each bounding box. The dataset we collect consists of four classes of fruits: Apple, banana, orange, pear. We have 1,690 images in total, the sample number of each class is 400. This dataset was categorized into three groups for training, validation, and testing. Table 1 shows the proportion of our dataset.

Table 1. The numbers of digital images in the dataset

Description	Total	Training	Validation	Testing
Quantity	1,690	1,352	169	169

This dataset for fruit detection was collected by using the camera of a mobile phone with the resolution of 1920 × 1080, nearly 3,000 images. Finally, we selected 1,690 images from those photos. In our dataset, there is a single fruit in each image or multiple ones in the same image. That could test the ability of our model for conducting single object detection and multiobject detection.

4.2 Experimental Results

From our experiments, the performance of CenterNet with different backbone nets is desirable by using our dataset, we make use of loss functions to observe the convergence

Fig. 5. The samples of our dataset

of model training. The loss function of CenterNet includes three parts: Heatmap loss or center loss, offset loss, and size loss.

Both DLA-34 and Hourglass have a good performance at the speed of convergences. From observation of the structure, DLA-34 not only has the IDA to connect the different layers to transform the feature information from bottom to top like what the ResNet does, but also merges the feature from multiple branches and scale. In this way, it immensely enhances the ability to combine various features to reduce redundant computation. After compared to the convergency of our models based on different backbones, the time costs of each model are listed in Table 2.

Table 2. The time cost of each model (seconds)

	Loading	Preprocessing	Object detection	Post-processing	Total
DLA-34	0.001	0.006	0.047	0.001	0.055
ResNet-18	0.001	0.008	0.017	0.001	0.028
Hourglass	0.001	0.008	0.046	0.001	0.056

All these three models have a similar distribution of time costs. The time spent on loading, preprocessing and postprocessing does not reach half of the whole time. The most consuming part is object detection, there are lots of convolutional neural operations in the step. Thus, we conclude that the model based on ResNet-18 is thought of as the fastest model.

Compared to these three methods, the result of DLA-34 is outperformed than other ResNet and Hourglass as shown in Table 3. It has the highest mAP 0.9, which is evidently higher than ResNet-18, a little bit better than Hourglass. The performance of DLA-34 in precision and recall is better than ResNet and Hourglass.

The reason why DLA-34 and Hourglass converge very fast is the well-designed network structure. Both of them have a much complex structure to deal with the information which got from various branches, not only limited by the depth.

Table 3. The average accuracy analysis

	DLA-34		ResNet-18		Hourglass	
	Precisions	Recalls	Precisions	Recalls	Precisions	Recalls
1	0.870	0.744	0.781	0.665	0.845	0.722
2	0.995	0.898	0.953	0.816	0.996	0.878
3	0.987	0.899	0.914	0.818	0.975	0.877
4	0.801	0.845	0.655	0.697	0.717	0.765
5	0.878	0.906	0.793	0.827	0.855	0.887
mean	0.906	0.858	0.8194	0.765	0.878	0.826
mAP	0.9		0.786		0.88	

DLA-34 takes use of the aggregation module to combine the feature from different depth layers, at the same time, DLA-34 also uses the HDA module to combine the feature from other scales by using upsampling. In this way, it reuses and integrates the feature information from various layers, just like the method of ResNet.

As a typical encoder to decoder structure, Hourglass merges the information from multiple scales like DLA-34. In the downsampling part, it extracts visual feature from input images in multiple scales. Then, the features are merged in the upsampling part. Hence, it efficiently merges and reuses the feature information, which is alike to the DLA-34. In a word, DLA-34 and Hourglass have a better performance in the convergency than the ResNet-18 due to the advanced structure.

After the analysis of the performance in convergency, we know the structures of DLA-34 and Hourglass are more complex than that of ResNet. Throughout the complex structures, DLA-34 and Hourglass could have better results of accuracy than ResNet.

For object detection, we make a tradeoff between speed and accuracy. Therefore, by comprehensively analyzing the experimental results, in this project, we chose DLA-34 as the backbone of our CenterNet model to settle this fruit detection problem.

Figure 6 shows the image samples for CenterNet based on DLA-34. From Fig. 6, we see that those full-view fruits should be detected from an image perfectly. Even if there are several fruits at the corner, our model is still able to detect them easily.

Figure 7 shows the examples of object detection from digital images. Those samples reveal that our model is able to detect both single object and multiobject with high accuracy. Our model still is able to detect them from the given images. From this work, we find the performance of our network is quite well. It is able to not only detect the class of our fruit correctly but also mark those fruits appropriately.

As a one-stage detector without anchor, the accuracy of the anchor-free detector is rather high based on our dataset. We detect a single fruit or multiple fruits together. For those occluded objects, our model still is able to detect the fruits which are overlapped.

Fig. 6. The successful examples for fruit detection based on DLA-34

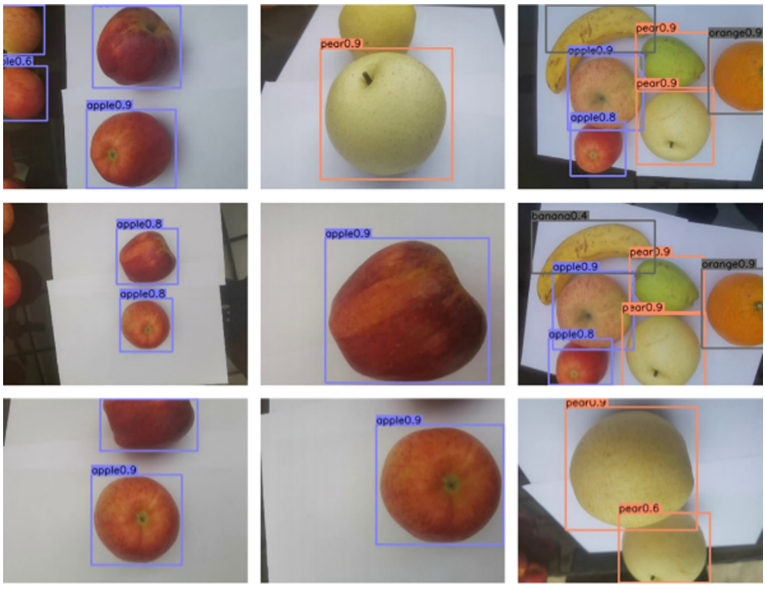

Fig. 7. The examples of object detection result of CenterNet based on DLA-34

5 Conclusion

In this paper, we set forth fruit detection from digital images. We prob the object detection problem based on computer vision, which is mainly comprised of two branches: ML-based methods and DL-based methods. We briefly introduce the ML-based methods. Then, we reviewed the DL-based methods. We concisely present the innovation and shortcoming. Finally, CenterNet is used as our proposed model. After designed the algorithm to conquer the problem of easy sample dominant and collecting dataset, we choose three networks: DLA-34, ResNet-18, and Hourglass as our backbone to train our model. Based on the results of our experiments, by taking into consideration of convergency, speed, and accuracy, the DLA-34 method is finally chosen as our backbone to tackle this fruit detection problem. In a nutshell, the CenterNet based on DLA-34 is the best method that is able to successfully handle our fruit detection problem.

In future, a more powerful backbone and training platform should be accommodated to design a better model [31, 32]. This will assist us to design and compare those models more quickly and easily for fruits and food science [35–40]. In future, we will apply a newer model to achieve a better result for this fruit objection problem [33, 34, 41].

References

1. Edan, Y., Han, S., Kondo, N.: Automation in agriculture. In: Springer Handbook of Automation, pp. 1095–1128 (2009). https://doi.org/10.1007/978-3-540-78831-7_63
2. Moltó, E., Pla, F., Juste, F.: Vision systems for the location of citrus fruit in a tree canopy. J. Agric. Eng. Res. **52**, 101–110 (1992)
3. Voulodimos, A., Doulamis, N., Doulamis, A., Protopapadakis, E.: Deep learning for computer vision: a brief review. Comput. Intell. Neurosci. **2018**, article ID 7068349 (2018). https://doi.org/10.1155/2018/7068349
4. Prince, S.J.: Computer Vision: Models, Learning, and Inference. Cambridge University Press, Cambridge (2012)
5. Nixon, M., Aguado, A.: Feature Extraction and Image Processing for Computer Vision. Academic Press (2019)
6. Gould, S.: DARWIN: a framework for machine learning and computer vision research and development. J. Mach. Learn. Res. **13**(1), 3533–3537 (2012)
7. Viola, P., Jones, M.: Robust real-time object detection. Int. J. Comput. Vision **4**, 34–47 (2014)
8. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE CVPR 2005, pp. 886–893 (2005)
9. Felzenszwalb, P.F., Girshick, R.B., McAllester, D.: Cascade object detection with deformable part models. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2241–2248 (2010)
10. Rosenfield, M.: Computer vision syndrome: a review of ocular causes and potential treatments. Ophthalmic Physiol. Opt. **31**(5), 502–515 (2011)
11. Patrício, D.I., Rieder, R.: Computer vision and artificial intelligence in precision agriculture for grain crops: a systematic review. Comput. Electron. Agric. **153**, 69–81 (2018)
12. Rosenblatt, F.: The perceptron: a probabilistic model for information storage and organization in the brain. Psychol. Rev. **65**(6), 386 (1958)
13. Gomolka, Z.: Backpropagation algorithm with fractional derivatives. In: ITM Web of Conferences, vol. 21, p. 00004 (2018)
14. Werbos, P.J.: Backpropagation through time: what it does and how to do it. Proc. IEEE **78**(10), 1550–1560 (1990)

15. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Commun. ACM **60**(6), 84–90 (2017)
17. Hubel, D.H., Wiesel, T.N.: Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. J. Physiol. **160**(1), 106 (1962)
18. Fukushima, K.: Artificial vision by multi-layered neural networks: neocognitron and its advances. Neural Netw. **37**, 103–119 (2013)
19. LeCun, Y., et al.: Backpropagation applied to handwritten zip code recognition. Neural Comput. **1**(4), 541–551 (1989)
20. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: DropOut: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
21. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
22. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the Inception architecture for computer vision. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
23. Girshick, R.: Fast R-CNN. In: IEEE International Conference on Computer Vision, pp.1440–1448 (2015)
24. 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, pp. 91–99 (2015)
25. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 779–788 (2016)
26. Liu, W., et al.: SSD: single shot multibox detector. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 21–37. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_2
27. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. arXiv:1904.07850 (2019)
28. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
29. Yu, F., Wang, D., Shelhamer, E., Darrell, T.: Deep layer aggregation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2403–2412 (2018)
30. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9912, pp. 483–499. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46484-8_29
31. Yan, W.: Introduction to Intelligent Surveillance - Surveillance Data Capture, Transmission, and Analytics (Third Edition), Springer (2019). https://doi.org/10.1007/978-3-030-10713-0_1
32. Yan, W.: Computational Methods for Deep Learning - Theoretic, Practice and Applications. Springer (2021). https://doi.org/10.1007/978-3-030-61081-4_1
33. Pan, C., Yan, W.Q.: Object detection based on saturation of visual perception. Multimed. Tools Appl. **79**(27–28), 19925–19944 (2020). https://doi.org/10.1007/s11042-020-08866-x
34. Pan, C., Yan, W.: A learning-based positive feedback in salient object detection. In: IVCNZ (2019)
35. Al-Sarayreh, M., Reis, M.M., Yan, W.Q., Klette, R.: Detection of adulteration in red meat species using hyperspectral imaging. In: Paul, M., Hitoshi, C., Huang, Q. (eds.) PSIVT 2017. LNCS, vol. 10749, pp. 182–196. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75786-5_16

36. Al-Sarayreh, M., Reis, M., Yan, W., Klette, R.: Detection of red-meat adulteration by deep spectral-spatial features in hyperspectral images. J. Imaging **4**(5), 63 (2018)
37. Al-Sarayreh, M., Reis, M., Yan, W., Klette, R.: Chemometrics and hyperspectral imaging applied to assessment of chemical, textural and structural characteristics of meat. Meat Sci. **144**, 100–109 (2018)
38. Al-Sarayreh, M., Reis, M.M., Yan, W.Q., Klette, R.: A Sequential CNN approach for foreign object detection in hyperspectral images. In: Vento, M., Percannella, G. (eds.) CAIP 2019. LNCS, vol. 11678, pp. 271–283. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29888-3_22
39. Al-Sarayreh, M., Reis, M., Yan, W., Klette, R.: Deep spectral-spatial features of snapshot hyperspectral images for red-meat classification. In: IEEE IVCNZ (2019)
40. Al-Sarayreh, M., Reis, M., Yan, W., Klette, R.: Potential of deep learning and snapshot hyperspectral imaging for classification of species in meat. Food Control **117**, 107332 (2020)
41. Liu, Z., Yan, W., Yang, B.: Image denoising based on a CNN model. IEEE ICCAR **1**(1), 389–393 (2018)