Fruit Detection Using CenterNet

Kun Zhao

A thesis submitted to Auckland University of Technology
in partial fulfillment of the requirements for the degree of
Master of Computer and Information Sciences (MCIS)

2020

School of Engineering, Computer & Mathematical Sciences

Abstract

In today's world, agriculture automation is becoming more and more important. This thesis is based on how to detect fruits from digital images. The relevant approaches are split into two parts: Machine learning-based methods and deep learning-based methods. After introduced those methods and comparing different deep learning-based methods, CenterNet is chosen as our model to settle this fruit detection problem. Three CenterNet models with different backbones were introduced, the backbones are ResNet-18, DLA-34, and Hourglass. A fruit dataset with four classes and 1,690 images was collected for this research work. By comparing those models with different backbone, according to the results, the deep learning-based model with DLA-34 was chosen as the final model to detect fruits from an image, the performance is excellent.

In this thesis, the contribution is that we deploy a model based on CenterNet for object detection to settle the problem of fruit detection. Meanwhile, our dataset with four classes and 1,690 images were collected. By evaluating the performance of the model, we eventually design a CenterNet based on DLA-34 to detect multiclass fruits from our images. The performance of this method is better than the existing ones for fruit detection.

Keywords: CenterNet, ML-based method, DL-based method, fruit detection, agricultural automation, ResNet-18, DLA-34, Hourglass net

Table of Contents

List of Figures

List of Tables

Attestation of Authorship

I hereby declare that this submission is my work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgments), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Signature: Date: 15 November2020

Acknowledgment

First of all, I would like to express my thankfulness to my family. Thanks for their support and help, so that I can focus on this thesis during my time of academic study in New Zealand. On the second, I want to express my thanks for AUT. Thanks to the facilities and resources that they support me. I also thank the collaborations in my research group and laboratory. Thanks to their teamwork which helps me get improvement.

My deepest thanks should be given to my supervisor Wei Qi Yan, not only for his practical guidance and advice, but also for how to learn, how to think, and how to work. I think I could not finish this thesis and achieve my Master's degree without his help and supervision. Besides, I would like to appreciate the AUT administrators for their support and guidance in the past years.

Kun Zhao

Auckland, New Zealand

15 November 2020

Chapter 1
Introduction

In this chapter, we will introduce the background of fruit detection in agriculture automation. Meanwhile, we will interpret the motivation of why we commence this research project. After that, we review the existing work related to object detection, our method about how to handle this problem also will be demonstrated. Furthermore, the contributions of this thesis will be presented. The structure of this thesis will be stated in the latest part.

1.1 Background and Motivation

Agriculture automation is a crucial role in modern industry, which could produce more output to feed the increasing global population. The fruit industry, as a typical one with high economic value, has an intensive requirement for automation (Edan, Han, & Kondo, 2009).

In the fruit industry, the 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 (Hayashi, Ueda, & Suzuki, 1988).

For harvesting robots, several tasks need to be handled, such as detection, picking, localization, classification, selection, and grading. Among these missions, object detection is the most critical one, hence, we should settle this problem first (Moltó, Pla, & Juste, 1992).

It is natural to find several fruit objects from the environment for our human beings. However, it is not easy for robots to tackle this problem. For robots, the first mission is how to know the environment. Generally, the most important information is the image part. Various methods could be used to obtain images from outside (Xin, & Shao, 2005). In the past decades, multiple methods are having been used to obtain an image. Those methods are based on digital cameras to obtain various images, such as universal camera, depth camera, near-infrared camera. In recent years, a universal camera of our cellphone achieved a great improvement in speed and resolution (Lowe. 2004). Hence, because of the cost, portability, resolution, a universal camera is used to attain images from the picking field in our research (Schalkoff, 1989).

The following work shows how to detect visual objects from an image, which is thought of as an answer to object detection in computer vision (Moltó, Pla, & Juste, 1992). The methods used to handle this research problem mainly have been split into two groups: machine learning-based (ML-based) method and deep learning-based (DL-based) method

(Voulodimos, Doulamis, Doulamis, & Protopapadakis, 2018).

Because of those methods, in this research project, we want to design a model to settle this fruit detection problem. What the ML-based methods as the base of visual object detection give are the main process of how to deploy a model for detection questions during the past research. However, those methods have their limitations due to the development of the theory and hardware in the past decades (Szegedy, Ioffe, Shlens, & Wojna, 2016). DL-learning methods will be discussed in the second part, eventually, we design a DL-based model that could settle fruit detection fast and accurately.

In summary, fruit harvest robots are used to effectively improve the output of the fruit industry and save human labor. The base is that the robot could find fruits from the outside environment fast and accurately. There are both ML-based and DL-based methods could be used to handle this problem. This thesis will talk about both of them in advantage and disadvantage and give our model to process this problem.

1.2 Research Questions

This research project aims to tackle how to use object detection methods for fruit detection. Therefore, the research questions of this thesis are:

(1) What are the existing ML-based methods that can be used to tackle object detection questions?

(2) How to settle this object detection problem by using DL-based methods?

(3) Could DL-based resolve this problem accurately and rapidly?

The main purpose of this research is to find a practical model that could resolve this problem with object detection for fruit. Therefore, we will discuss the existing ML-based methods. Consequently, we detail the advantages and disadvantages. Then, the DL-based methods will also be briefed. Finally, we choose the most suitable method for this research 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 dataset.

1.3 Contribution

The focus of this project is on how to detect fruit from an image by using deep learning, which could be thought of as an object detection problem. Hence, this thesis will discuss the existing methods of object detection, namely, both ML-based and DL-based methods. Meanwhile, those methods will be evaluated and compared. Eventually, CenterNet is regarded as an effective method for this project.

Moreover, it will be tested based on three different backbone models and pick up the most suitable one. In this research project, we also collect a new dataset for four classes of fruits. Besides, we train and test the model by using our dataset. Eventually, in this thesis, we present an evaluation of how CenterNet works by using this dataset. The contributions of this thesis are:

- Design a model to handle fruit detection problems based on deep learning-based methods, CenterNet.
- A dataset with four classes of fruits for this research project has been created.
- CenterNet will be tested based on three different backbone models, and the performance is assessed.
- Training and testing this model by using our dataset, as well as, evaluating the performance of this model in this thesis.

1.4 Objectives of This Thesis

In this thesis, our objective is to clarify the background of fruit detection. Moreover, we also need to detail the existing methods and find the most suitable one for our project. Furthermore, we need to collect image data for our model to confirm that can resolve this problem. Our objectives are summarized as:

- Review the existing methods for object detection and find the most potent model for fruit detection.
- Collect our data to training and testing this model, finally evaluate its performance

for fruit detection.

- Design our model to ensure that it is able to detect fruits from digital images accurately and rapidly.

1.5 Structure of This Thesis

The structure of our thesis is as:

- In Chapter 2, we will discuss the related work that has been done by forerunners. We also make an overview of object detection based on computer vision. From two typical branches of computer vision, machine learning-based methods and deep learning-based methods. Finally, we will confirm a suitable method as our model to settle this fruit detection problem.
- In Chapter 3, we will talk about the methodology of our research. Our dataset for fruit detection will be introduced, and how to use it to fulfill the detection problem consequently will be explained. Besides, we will also explicit the tools we will use in this research project. Then, how to design our net by using three different backbone nets and how to evaluate it will be discussed at the end of this chapter.
- In Chapter 4, the result of our model will be shown and the performance will be discussed in three aspects: Convergency, time cost, and accuracy analysis. This model is designed with three different backbone nets. Finally, we will make a summary of all of the three models.
- In Chater 5, the most suitable model will be chosen. After this, we will discuss the results. The advantages and disadvantages will be expounded in this chapter. Finally, we will give the limitation of our research work.
- In Chapter 6, we will draw our conclusion and illustrate the potential directions as well as improvements for future work.

Chapter 2
Literature Review

This chapter will explain the related work that had been conducted in object detection. There are a vast number of research results that have been achieved in the last decades. Those methods generally come from computer vision (CV), from the previous machine learning-based methods to the deep learning-based methods. Hence, we will illustrate the achievement that those forerunners have attained. Finally, we will also propose the method of ours and explain how we deal with this problem.

2.1 Introduction

Computer Vision (CV) is a subject to investigate the visual ability of computers (Forsyth, & Ponce, 2002). The main tasks are object classification, object locating, object detection, object tracking, object segmentation, etc. (David, 2004). Regarding this research project, the task that we need to work is how to detect fruits from the given images, which is thought of as a typical object detection problem (Nixon, & Aguado, 2019). Hence, this research project will be conducted especially for fruit detection.

A great deal of pioneers have worked for vast plenty of research around object detection. A large number of methods have been designed and implemented. Those methods are grouped twofold depending on a theoretical basis. The first one is the machine learning-based (ML-based) methods (Sebe, Cohen, Garg, & Huang, 2005), the second one is the deep learning-based (DL-based) methods (Prince, 2012). In the following sections, the history, purpose, advantages, and disadvantages of both ML-based methods and DL-based methods will be detailed.

2.2 Machine Learning-Based Methods

Machine Learning (ML) was defined as a kind of method (Samuel, 1959), which means that ML models have been trained to implicitly resolve a given task, whose procedure does not need human intervention and completely is based on a tangible algorithm. The ML was defined again (Blum & Mitchell, 1998) as a computer program, which can learn from experience E for task T, its result can be improved by using performance P (Blum & Mitchell, 1998).

ML inspires us in multiple ways, especially for pattern classification from huge amounts of high-dimensional images (Nixon, & Aguado, 2019), which also brings the development of computer vision (Gould, 2012). The typical progress of the ML-based methods is shown in Figure 2.1 (Papageorgiou, & Poggio, 2000).

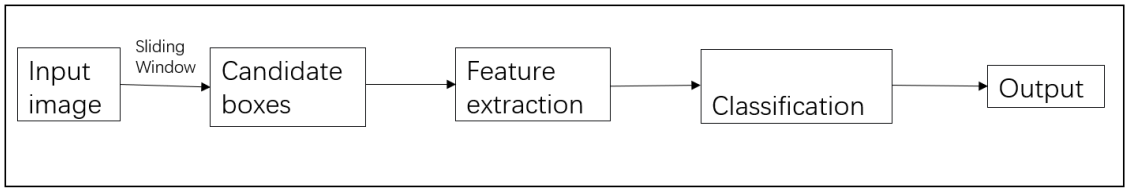

Figure 2. 1 A pipeline of ML-based method for CV

In Figure 2.1, we understand the pipeline of the Ml-based method for object detection. After obtained an image, the sliding window method will be applied to this image (Laguna, Olaya, & Borrajo, 2011), then we receive lots of candidate bounding boxes. This process is named region selection (Cho, & Tropsha, 1995), which means the region we want to detect has been selected into these boxes. The consequent image is an example of this progress.

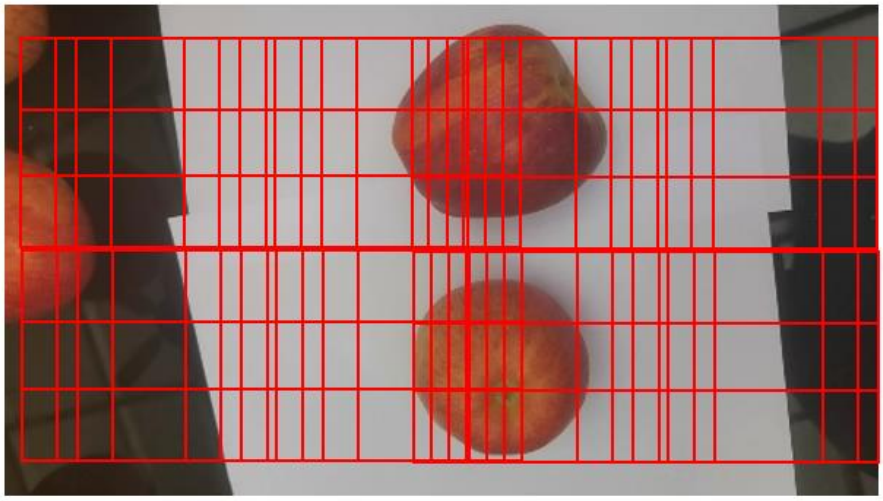

Figure 2. 2 An example of the sliding window method

In Figure 2.2, we directly find that this method will generate a myriad of candidate boxes for one image, those boxes could be of different sizes and shapes. After this step, feature extraction will be implemented for each candidate bounding box (Llorca, Arroyo, & Sotelo, 2013). Then, the feature inside each box will be extracted from the image by using specific methods. Consequently, a classifier will be used to classify each box based on its features. Finally, what class this box is will be given in the last step.

In 2004, Viola and Jones put forward a method named VJ method to detect a specific object from an image (Viola, & Jones, 2014). The working progress is subject to the ML-based method that we just mentioned. It uses the Haar feature (Lienhart, & Maydt, 2002) as their target feature to conduct the feature extraction. Then, it takes AdaBoost (Rätsch, Onoda, & Müller, 2001) as the classifier to classify each box based on the previous extracted Harr feature. Eventually, what class the box is will be given by this AdaBoost classifier.

In 2005, a method, which was based on SVM (Suykens, & Vandewalle, 1999) and HOG (Dalal, & Triggs, 2005), was brought up by Dalal and Triggs (Dalal, & Triggs, 2005). The feature is named HOG, which means a histogram of oriented gradients that could express complex features by combing several different features into a multi-dimension feature (Xie, Liu, & Qu, 2009). Instead of AdaBoost, this method uses SVM as their classifier to classify each box, which achieves a better performance than the VJ method (Suykens, & Vandewalle, 2006).

After the method of Dalal and Triggs, a new method named DPM was put forward in 2008 (Felzenszwalb, McAllester, & Ramanan, 2008). The working progress of DPM is still subject to ML-based methods. Similarly, the classifier of DPM is SVM, yet (Felzenszwalb, Girshick, & McAllester, 2010). The change which makes it has better performance is the feature extraction (Ghiasi, & Fowlkes, 2014), which enhances the HOG feature by using a combing signed gradient with an unsigned gradient to make it a richer one to express a broad spectrum of visual objects (Felzenszwalb, & Ramanan, 2008). Meanwhile, it also utilizes PCA (principal component analysis) to work for dimension reduction to reduce complexity and accelerate the speed (Borji, Cheng, Jiang, & Li, 2015). Hence, it could get a balance between the complexity and the speed of this model.

In Table 2.1, we make a summary of ML-based methods based on these three models. We find all of them are subject to the progress shown in Figure 2.1. The method of region selection is the same by using a sliding window method. But the feature which is used to

be classified differs and the classifier that is used to classify the objects get improvement along with the development of CV (Blehm, Vishnu, Khattak, Mitra, & Yee, 2005). Even DPM had got a very good performance in 2005, but there are still several remaining problems for the ML-based methods (Han, Shao, Xu, & Shotton, 2013).

Table 2. 1 The summary of ML-based methods

Method	Region Selection	Feature	Classifier
VJ	Sliding Window	Haar	AdaBoost
SVM+HOG	Sliding Window	HOG	SVM
DPM	Sliding Window	HOG + PCA	SVM

Regarding ML-based methods, the features generally are extracted for a particular object (Brosnan, & Sun, 2004). Because of this fact, the visual features for object detection generally are various. In another word, those features are not transportable, which means the designer needs to specify visual features for different objects (Rosenfield, 2011).

Apart from the feature design, we still need training and testing this model separately and repeatedly (Patrício, & Rieder, 2018). The reason is that the different part of the same model has distinctive functions, the information will not be shared between them. Besides, the sliding window method will generate lots of bounding boxes to detect the visual object. Even if it is useful, the percentage of the background is considerably high in those boxes. There is not an object inside those boxes, the computing for them makes no sense. However, they still need to waste considerable computation to be computed.

In summary, along with the development of CV, a plethora of methods have been implemented to settle object detection problems. The typical three ML-based methods are the VJ method, SVM + HOG method, and DPM method. They have better performance for object detection, which should attribute to the improvement of the feature extraction methods and better classifiers. However, as the inherent problem of the ML-based methods, the disadvantages such as private, complexity, and high computation limit the usage and development of these methods.

2.3 Deep Learning-Based Method

In this project, we know that ML-based methods could be applied to resolve the object detection problem. However, complexity limits its usage and development. But with the improvement of CV and DL, a plenty of methods for object detection have been implemented after DPM. Those methods are based on deep neural networks (Szegedy, Toshev, & Erhan, 2013), especially for the convolutional neural network (CNN).

As the origin of CNN, the relevant work of artificial neural networks (ANNs) will be discussed first. CNN as the deepened development of ANNs will be presented following. Afterward, the representative methods based on CNN having good performance for the detection problem will also be introduced. Even though those methods have their limitations, the emerge of these methods greatly inspired the research work in this area.

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. This kind of model automatically extracts features from the training data by using a specific algorithm in the end-to-end way (LeCun, Bengio, & Hinton, 2015).

2.3.1 Neurons

The ANNs were inspired by research outcomes of our human brain, which deals with various tasks by simulating the mechanism of the human neurons. The structure of a neuro chiefly consists of dendrite, axon, and cell nucleus. One nerve cell has many dendrites,

which are used to receive the impulse from the nerve endings of multiple previous nerve cells (Thompson, etc. 2001). Then, those input impulses will be accumulated by using the cell nucleus. When the sum of the input signals surpasses a threshold, the cell nucleus will have stimulation and generate an electronic signal. These electronic signals will be transmitted through the axon and are sent to other nearby nerve cells. Our human neural system contains near 86.9 billion neural cells like that. Those cells and their connections form huge complex networks (Azevedo et al. 2009).

Furthermore, even each single nerve cell in our human brain is not important, but the essential part is how those neural establishes a complicated network. In this way, the performance has a drastic improvement than just a simple structure. The key point is that the connections between neural cells are various, which have different strengths. For those joint strengths, they will change along with the activation of the nerve cell (Pehlevan, & Chklovskii, 2015). Hebbian theory points out that the joint strengths between two nerve cells will enhance if they stimulate each other with a high frequency (Zador, Koch, & Brown, 1990). Followed this process, our short-term memory will be transformed into long-term memory.

2.3.2 Artificial Neural Networks (ANNs)

Inspired by the structure and the mechanism of our human brain neural system, a computing model named ANNs was proposed, which imitates our brain neural system by connecting lots of computing nodes.

The first and basic model was named the MP (McCulloch and Pitts) model, which is a kind of artificial neural cells based on simple computation (McCulloch, & Pitts, 1943). Its presence means the beginning of the research about ANNs. The base of the current deep neural network was named Perceptron in 1958 (Rosenblatt, 1958). The original purpose was to handle binary classification problems. The basic computing unit is the neuron, which simulates the working process of the human brain neural cell.

The typical structure contains the input unit x, its corresponding weights w and bias b, the output units, and the processing unit. Given the quantity of input is 3, this neuron

will compute the product of w and x, then add a bias and calculate the sum of all three results. Finally, an activation function $f(\cdot)$ will be used to process the value of the sum, which will obtain one result for two classes.

LMS (Least Mean Square) is a method for measuring the difference between the truth data and the predicted results of perceptron (Widrow, & Lehr, 1990). In 1967, SGD (stochastic gradient descent) was used to train the network by reducing the error of LMS, which made the perceptron to be more intelligent (Amari, 1993).

The promotion of perceptron could be thought of as an improvement of ANNs. However, it is not enough for dealing with various tough structures, such as xor circuit problems (Latypova, & Tumakov, 2018). One reason is that the activation function of the perceptron is merely a step function, just like equation (2.1).

$$step(x) = \begin{cases} 1, & x > threshold \\ 0, & otherwise \end{cases}, \tag{2.1}$$

For only one perceptron, it could not deal with the nonlinear problem, such as the above-mentioned xor problem. Meanwhile, limited by the development level of the computer industry, the requirement of computations is quite huge, which could not be satisfied at that time (Minsky, & Papert, 2017).

In 1970, automatic differentiation was put forward, which is the base of BP (Back Propagation), which is the crucial method for improving the speed of training of ANNs (Gomolka, 2018). After this, Werobs suggested the formal idea that combing BP and ANNs, which inspired the subsequent MLP (Multilayer Perception) (Werbos, 1990). This important model was deployed in 1986. This new model is not a single neural cell, which is a multilayer net with a middle layer named hidden layer (McClelland, 1986). Furthermore, the activation function was replaced by a sigmoid function

$$sigmoid(x) = \frac{1}{1 + e^{-x}}. \tag{2.2}$$

This nonlinear map function could enhance the performance of MLP, which could effectively tackle the nonlinear classification problem (McClelland, etc. 1986), MLP is also named a shallow network.

The improvement of MLP is very huge and it overcomes the challenge of the perceptron. But, its performance is far worse than the human brain cell. According to the

structure of the human brain, the natural idea is to overlap more layers to imitate the multi-layer stacked human brain, rather than the shallow network. But how to train a deep network confuses researchers for a long time.

Hochreiter shows that gradients vanishing and gradients explosion are the reasons why deep nets could not be trained. The attenuation of error for BP in the deep net will exponentially increase or decrease following the increase of the network layer (Hochreiter, & Schmidhuber, 1997).

The breakthrough was in 2006, a network named deep belief nets (DBN) was designed (Hinton, etc. 2006). This network dealt with the hard challenge of training deep networks by using layer-by-layer pretraining. Firstly, a DBN was trained layer by layer. Then, the weights of this pre-trained DBN will have the initial specific weight of the forward neural network (FNN) rather than the random weight. Finally, these specific weights will be adjusted by using the BP method (Hinton, etc. 2006).

This network solved the problem that a DNN is hard to be trained by using layer by layer pre-training, which makes the ANNs could be used to tackle the tough problems in speech recognition, natural language processing (NLP), and image classification (Krizhevsky, Sutskever, & Hinton, 2017). It has a better performance than the ML-based method in object detection problems.

2.3.3 CNN

Convolutional neural network (CNN) as a kind of ANNs, was inspired by the visual system. The original conception was enlightened by the visual layer cells of cats, Hubel and Wiesel named it the receptive field (Hubel, & Wiesel, 1962). They found that, in the primary visual cortex of cats, the different neural cells have a different preference for the diverse light bands. The structure of our visual cortex is a hierarchy, the signal from the eyes will come to the primary visual cortex (V1). After the simple processing of V1 for some detail and direction, those signals will be transmitted to V2. V2 will deal with those edges and outline information and express it as some simple shapes.

The neural cell of V4 as a kind of color-sensitive cell will cope with these signals and

transmit them to Inferior Temporal (IT) Cortex. All information will be combined to express a complex objection (Arcaro, & Livingstone, 2017). In 1979, Neocognitron was put forward by Fukushima. This conception combined ANNs and Receptive Field, which was thought of as the first CNN structure (Fukushima, 2013). The work led to an important conception of the local connection of CNN (Feng, 2019). The local connection is also named partially connection, which means each node of the current convolutional layer only connects to the part of the previous layer. Just like each visual layer only processes the signals obtained from their previous layer.

In 1989, weight sharing as one of the main concepts of CNN was brought up by LeCun (LeCun, etc. 1989), which means each convolutional kernel will be used to detect a particular feature and greatly decrease the parameter quantity of CNN to make the complex computation to be possible.

In 1998, modern CNN was suggested, LeCun combines convolutional layers and downsampling layers to design a new model named LeNet. The main structure of LeNet is shown in Figure 2.3. The main idea of the convolutional part is shown in Figure 2.4.

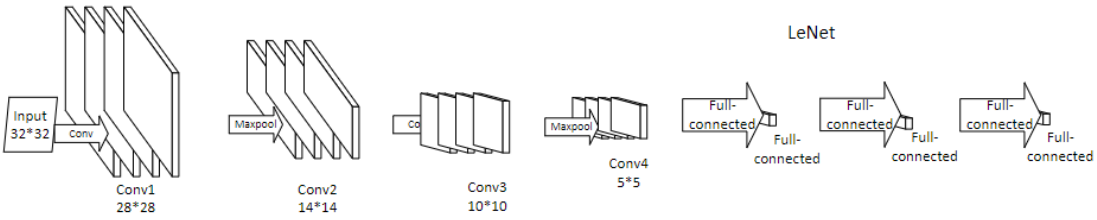

Figure 2. 3 The structure of LeNet

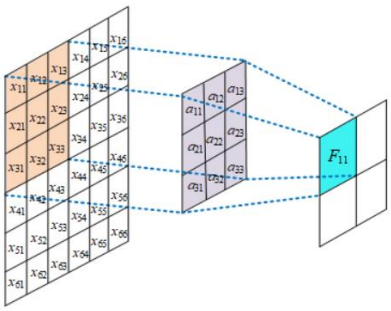

Figure 2. 4 A sample for a convolutional network

This is an example of a convolutional neural network, F is a result of the convolutional computation between input x and kernel a, F is also named as a feature map, which is thought as the result after the kernel extracts features from x. A specific feature is detected from input x. Each computation is regarded as a process workflow of the visual neural layer, which processes the information from the previous layer then passes it to the next layer (LeCun, etc. 1998).

2.3.4 AlexNet

As the consequent model of GooLeNet, AlexNet was put forward in 2012 (Hinton, 2012). In 2012, AlexNet obtained the champion of ILSVRC (ImageNet Large Scale Visual Recognition Challenge). Its performance is far better than other methods, this result means the DL-based method has transcended the ML-based method.

The reason why AlexNet has considerable progress is due to its various improvement. Firstly, it replaces the sigmoid function with a new activation function. The sigmoid function could be used to make each unit to be nonlinear in the initial part of ANNs. However, whether the input is extremely big or small, the gradient will be 0, which will lead to a gradient disappear problem (Hinton, Srivastava, & Swersky, 2012). In order to settle this problem, ReLU (Rectified Linear Units) is deployed in AlexNet.

$$ReLU(x) = \max(0, x). \tag{2.3}$$

For the ReLU function, when the input is considerably huge, the gradient will not be 0 again. It could tackle the gradient vanishing problem by transporting the gradient to a deeper network. Furthermore, the computation of ReLU is easier than sigmoid, which could make the whole network converge more quickly.

Another improvement of AlexNet is to decrease overfitting by using the dropout method. Overfitting is a typical problem for both ML-based methods or DL-based methods, the performance is that the result for the training dataset is good while extremely bad for a test dataset (Hawkins, 2004). The dropout method could be used to tackle the

overfitting problem. The idea of the dropout method is randomly set some neural cells to stop working by a particular probability. In this way, it could make the whole network a better generalization ability that does not depend on local features (Srivastava, etc. 2014).

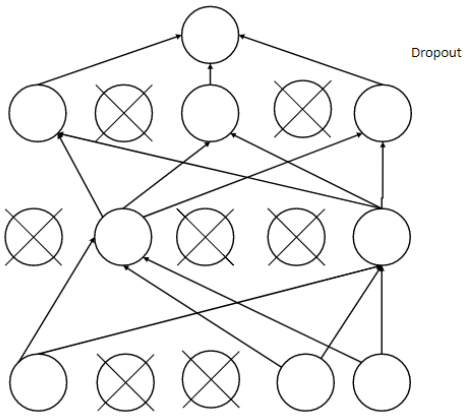

Figure 2. 5 A sample for dropout

In Figure 2.5, we find that multiple neural cells are closed while others still keep working. For the whole training process, the final result will combine all disadvantages or advanced fitting to obtain a balanced result.

2.3.5 VGGNet

VGGNet as the offspring network of AlexNet has several improvements to obtain a better performance in 2014 (Simonyan, & Zisserman, 2014). 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. The receptive field of two 3×3 kernels equals one 5×5 kernel. But the number of weight decreases from 25 to 18. For a bigger kernel, such as 7×7, it equals three 3×3 kernels. The number of weight decreases from 49 to 27. Meanwhile, more layer kernels mean that more activation functions, which will make the network could increase the learning ability due to a stronger nonlinear ability.

In this way, small-size kernels to replace a big kernel could efficiently decrease the number of weight and enhance the express ability of the whole network. Hence, the success of VGGNet shows us that a deeper network could bring much better performance, using a smaller size kernel could decrease the computation while keeping the performance

of the network.

2.3.6 GoogLeNet

For VGGNet, a deeper network brings better performance. But, it will also need more data to be trained, otherwise, it will lead to overfitting and gradient vanishing problems. Different from increasing the depth of the network, in 2014, GoogLeNet by using the inception module to tackle this problem (Szegedy, Ioffe, Vanhoucke, & Alemi, 2016).

The inception module is different from VGGNet to deepen the network that broadens the network. This module could parallelly execute multiple-size convolutional computation, then those feature regions will be concatenated together. Furthermore, the 1×1 kernel will reduce the parameters by using it to apply dimension reduction.

Thus, Inception v2 imitates VGGNet to use two 3×3 kernels to replace a 5×5 kernel (Szegedy, Vanhoucke, 2016). This could let the network have better performance and reducing more parameters.

2.3.7 R-CNN

ML-based methods have an accessible process for object detection. The procedure consists of three parts: Region selection by sliding window method, feature extraction, and classification for each bounding box, the result will be given out. Even if it could settle object detection problems, this kind of method has inherent problems.

The sliding window method could generate the object bounding box after region selection. But, its complexity and redundancy limit its speed and efficiency due to a huge computation, this restricts its transportability.

In 2014, a DL-based method named R-CNN (Girshick, et al., 2014) was implemented to overcome those problems. For one object, its visual information, such as texture, shape, color, and edge, could be similar and relevant. Hence, the region proposal method instead of the sliding window method can help us to get the candidate box.

The feature extraction of R-CNN takes use of the convolutional neural network (CNN) instead of the HOG feature. Moreover, AlexNet (Krizhevsky, Sutskever, & Hinton, 2012) is chosen to extract the feature of each box. By using AlexNet, this model could extract and learn features from boxes automatically. Eventually, AlexNet will generate a feature map for each box. Based on the designed features, the DL-based method overcomes the disadvantage of the ML-baed method in the feature extraction.

For the classification, R-CNN still chooses SVM as its classifier. SVM will classify each box based on its feature map, which was got from AlexNet. Through those changes, R-CNN will generate 2,000 candidate boxes from an image by using the region proposal method. Then those boxes will be extracted features by using AlexNet to generate 2,000 feature maps. Meanwhile, the SVM classifier will be conducted to classify each box, as well as a regressor to regress the bounding box. Because of its complex calculation, the speed of R-CNN is very slow.

2.3.8 Fast R-CNN

To overcome the disadvantage of R-CNN, Fast R-CNN was proposed by Girshick in 2015 (Girshick, 2015). The training process of R-CNN does not continue, which needs to train AlexNet and SVM separately. Firstly, it trains AlexNet in a big dataset, then fine-tuning makes it extracting the feature from our dataset. After this, it trains SVM for each class and regresses the bounding box. Hence, the complex progress confines its performance.

Fast R-CNN imitates R-CNN, but it does not use a lot of CNN networks to extract features from each region proposal. It directly uses VGG (Simonyan & Zisserman, 2014) to extract features from the input image, and generate only one feature map. Meanwhile, it will also conduct region selection to fetch region proposals based on the input image. In the next step, this feature map, the result of VGG, could be partially selected depended on the region proposal. Then, those selected feature maps will be warped and be reshaped into a fixed size by an ROI pooling layer. Finally, those wrapped feature maps will be sent into a fully connected layer and be prepared for using softmax to classify it and using

a linear regressor to regress the bounding box.

Hence, Fast R-CNN only conducts feature extraction one time based on the whole image, instead of abundant region proposals. This could save lots of time of computations and memory for saving the CNN node. This CNN model could share the feature they learned, instead of R-CNN, those private features.

On the other hand, Fast R-CNN could train classifiers and regressors at the same time, instead of R-CNN style. SVM, the classifier of R-CNN, also was changed into softmax.

2.3.9 Faster R-CNN

Faster R-CNN has a better performance than Fast R-CNN by using one CNN for the image instead of lots of CNN for region proposal. It also could train classifiers and regressors together. But the method to generate region proposal, selective search, still speeds lots of time. Therefore, how to overcome this bottleneck is the improvement of Faster R-CNN (Ren, He, Girshick & Sun, 2015).

The first step of Faster R-CNN is as same as Fast R-CNN, both of them are using CNN to generate a feature map. Different from Fast R-CNN, Faster R-CNN harnesses a neural net named RPN (Region Proposal Network) to replace selective search. This RPN net will use CNN to generate a lot of anchors with different sizes and ratios. Thus, this RPN will classify whether this anchor is foreground or background, meanwhile, it will also regress its bounding box. Finally, the RPN will discard lots of background information to reduce wasting time.

After the RPN net, Faster R-CNN will obtain a high-quality proposal, without abundant background. This will accelerate the speed of this model and increase accuracy.

2.3.10 YOLO

Faster R-CNN takes advantage of the RPN network to generate ROI. Then it conducts classification and regression with a high quality of anchor. Hence this process helps Faster

R-CNN get higher accuracy, but a slower speed. Therefore, instead of this two-stage method, a new one-stage method YOLO was designed to make the network very fast (Redmon, Divvala, Girshick, & Farhadi, 2016).

YOLO directly predicts the class and location of the object without anchor and RPN. Hence, its speed is very fast, but its accuracy has been reduced.

As the first one-stage object detection method, YOLO is also the first real-time detector. It does not have a prior box, the detection problem is thought of as a regression problem, hence its structure is very simple, which inputs an image with a size of 448×448, then after 24 convolutional layers and 2 fully connected layers, it will obtain a 7×7×30 feature map, which segments the input image into 7×7 regions, then predicts two boxes for each region. Hence, the number of all boxes is 98. If the center of one object falls in a region, this region will regress the ground box. It will remain the box with a bigger IoU score. Hence, YOLO v1 will detect 49 objections in total.

Those 30 channels are split into three parts, 20 probability of the class, 2 confidence values for boxes, and 8 location values for two boxes. Hence, it could detect 20 classes and 49 objects for each image (Redmon, 2016).

Through changing into a regression problem, YOLO v1 does not use the prior box and uses a GoogLeNet-like model as its backbone. These methods dramatically improve the speed of detection. But it also limits its performance in some small objects.

As the second version of YOLO, YOLO v2 also named YOLO9000 is able to classify 9000 classes, which makes use of DerkNet-19 as its backbone net and uses a prior box to detect the offset rather than size and location (Redmon, 2017). It also adopts multi-scale and multi-step training. The implementation of those methods brings in the improvement, however, it still not overcome the bad performance in small size object.

As the third version of YOLO, YOLOv3 takes in the attributes from the current detection frame, such as residual network and feature fusion. A network named DarkNet-53 was put forward (Redmon, & Farhadi, 2018).

By using the residual network, the network could be designed and developed very deeply. Meanwhile, the residual network could greatly alleviate the gradient disappears to accelerate the convergence. Through upsampling and concatenation, it will fuse both deep and shallow feature maps. Hence, it will generate three different size feature maps to be used for detection. Through different size feature maps, YOLO v3 has a better performance for small object detection even the speed is not very fast (Redmon, & Farhadi, 2018).

YOLO is the first real-time detector, which could achieve a balance for those scenarios which has a demand for speed. As a detector without an anchor, it also influences many anchor-free detectors.

2.3.11 SSD

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 (Liu, etc. 2016).

We find that it generates the different shapes of the feature map in fc6, fc7, conv8, conv9, conv10, conv11. Then, in SSD, anchors are generated for each feature map with different sizes and ratios. For the conv11, a big anchor is generated in SSD for the reason that this feature map has a huge receptive field, which could be applied to detect a big object. For the fc6, a small anchor is generated, the reason is contrary to that of conv11. The feature map has a small receptive field. Hence, a small size anchor could help SSD to detect very small objects.

Therefore, SSD combines the anchor of Faster R-CNN and directly predicts the result from YOLO. Furthermore, it makes use of multiscale feature maps to detect visual objects of various sizes. SSD does achieve better performance on speed and accuracy, However, the semantic information is not enough in the shallow feature map, this condition leads to difficulty for a small object.

2.3.12 CenterNet

The anchor is an important concept, which produces a huge improvement for Faster R-CNN, even SSD. But, it also brings several problems accompanying the benefit. For the hyperparameter of anchor, we need to design the number, size, height, width specifically. Those hyperparameters will influence our model directly. Besides, the anchor generally was uniformly sampled from the feature map. However, for our images, most of them just have the background, which needs a longer time to make sure whether the anchor is a positive sample or a negative sample.

Hence, multiple models were designed without the anchor. Inspired by the first anchor-free model, YOLO, a one-stage and anchor-free model named CenterNet was proposed in 2019 (Zhou, Wang, & Krähenbühl, 2019). It is designed based on CornerNet (Law, & Deng, 2018). CornetNet uses two corner points to predict that the object should be inside of these two corner points. However, it needs extra time to map those two corner points, and very easy to generate miss detection. Hence, CenterNet is a triplet instead of two key points. The structure of the CenterNet is shown in Figure 2.6.

CenterNet uses three heads to detect the 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 height, and width. This height and weight could be named embedding or size. The center gets from the feature map as a heatmap. In this heatmap, the location of the object has the highest value, which is the center. The center should be surrounded by embedding, which also uses offset to map the feature map to the original image.

Hence, this model used size, center, and offset to detect the object. As a kind of anchor-free model, we hope it could handle our fruit detection problem. Based on this, in this research, we choose CenterNet as our model to deal with fruit detection.

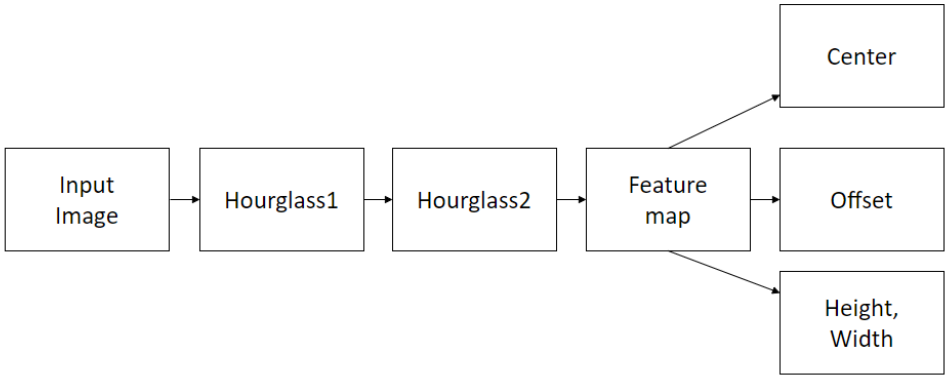

Figure 2. 6 The structure of CenterNet

Chapter 3
Methodology

The main content of this chapter is to explain research methods, which satisfy the objectives of this thesis. Hence, we will illustrate how to use CenterNet to detect fruits from a digital image. We will introduce the tools and the backbone module of this net.

3.1 Data Collection

Pertaining to this thesis, our task is to design a model and detect fruits from digital images. Therefore, we collect our dataset by taking photos of fruits firstly. Then, we label each fruit in each image to get the class and location of each bounding box.

The dataset we collect consists of four classes of fruit: Apple, banana, orange, pear. For the quantity of our dataset, we collected 1,690 images. For each class, its sample number is about 400. This dataset was categorized into 3 groups for training, validation, and testing. Table 3.1 shows the proportion of our dataset.

Table 3. 1 The proportion of the whole 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 photos, finally, we selected 1,690 images from those photos.

Figure 3. 1 The samples of our dataset

In our dataset, there is a single fruit in each image or multiple fruits in the same image. That could check the ability of our model for conducting single object detection and multiobject detection. Apart from this, there is a condition of overlapping in our dataset, such as the three images on the left side. As well as there are many fruits only partially in this image. For the same fruit, the angle of view is also different.

3.2 Data Labeling

Regarding this research project, the standard of our dataset follows PASCAL VOC 2007. This standard has been used in the main computer vision tasks, such as classification, detection, segmentation. In this thesis, what we should focus on is fruit detection. Hence, we will use two crucial folders: Annotations and JPEGImages. JPEGImages is the folder to store the original images. Figure 3.2 is an example of our JPEGImages folder, we store the image inside by using the .jpg format.

Figure 3. 2 The samples in the JPGEImages folder

Each image in this folder will be labeled by using a label tool named Labelimg. The information about the bounding box and class will be saved to an XML file, which is shown in Figure 3.3.

```xml
<annotation>
    <folder>JPEGImages</folder>
    <filename>1617.jpg</filename>
    <path>E:\python3\MST_apple\fruit_img\JPEGImages\1617.jpg</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>480</width>
        <height>270</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>pear</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>113</xmin>
            <ymin>41</ymin>
            <xmax>269</xmax>
            <ymax>198</ymax>
        </bndbox>
    </object>
    <object>
        <name>pear</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>266</xmin>
            <ymin>40</ymin>
            <xmax>385</xmax>
            <ymax>184</ymax>
        </bndbox>
    </object>
    <object>
        <name>orange</name>
        <pose>Unspecified</pose>
        <truncated>1</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>220</xmin>
            <ymin>191</ymin>
            <xmax>392</xmax>
            <ymax>270</ymax>
        </bndbox>
    </object>
</annotation>
```

Figure 3. 3 The sample of label data

From Figure 3.3, we find that there are three objects inside this image named 1617.jpg. Figure 3.3 shows that the class and the location of each box, such as the first object is a pear, the point of its left-top is (113, 41), its right-bottom corner is (269, 198).

3.3 Model Design

CenterNet is a one-stage and anchor-free model, which was chosen as our detection model. The structure of the CenterNet is shown in Figure 2.6.

From the net structure, we know that this model could locate the center of a visual object. The method of this progress is that it only finds the local peak point in the heat map. Each peak point means a center of an object which is found by our model, without NMS processing and anchor. It could save our running time. After obtained the peak point, the next features needed to be processed are the offset, height, and width of this object.

3.3.1 Algorithm Design

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. Then our model will map the information of the input image to the output feature map. This feature map will be transformed into a keypoint heatmap through Gaussian Kernal. 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 4 in our project. C is the number of classes. In our dataset, we only have four kinds of fruits, hence this C is 4. If 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.

Just as Figure 3.4 shows, the value of the most middle one is the most approximately near 1. It means there is an object here.

8	0.0164482	0.0271186	0.0397485	0.0517942	0.0599995	0.0617904	0.0565719	0.0460455	0.0333181	0.0214328
3	0.0348209	0.05741	0.0841475	0.109648	0.127019	0.13081	0.119763	0.0974783	0.0705344	0.0453733
3	0.0655342	0.108048	0.158368	0.206362	0.239054	0.246189	0.225398	0.183458	0.132748	0.0853941
5	0.109648	0.180779	0.264974	0.345273	0.399972	0.411911	0.377123	0.306951	0.222107	0.142877
	0.163096	0.268899	0.394133	0.513574	0.594936	0.612694	0.560949	0.456573	0.330372	0.212521
	0.21567	0.355579	0.521183	0.679126	0.786715	0.810197	0.741772	0.60375	0.436868	0.281028
	0.253538	0.418013	0.612694	0.798369	0.924849	0.952454	0.872015	0.709758	0.513574	0.330372
	0.264974	0.436868	0.64033	0.83438	0.966564	0.995415	0.911348	0.741772	0.536739	0.345273
	0.246189	0.405898	0.594936	0.77523	0.898044	0.924849	0.846741	0.689187	0.498689	0.320796
	0.203349	0.335266	0.491409	0.64033	0.741772	0.763913	0.699397	0.569259	0.411911	0.264974
3	0.149321	0.246189	0.360847	0.470201	0.544691	0.560949	0.513574	0.418013	0.30247	0.194573
5	0.0974783	0.160715	0.235564	0.306951	0.355579	0.366193	0.335266	0.272883	0.197455	0.127019
2	0.0565719	0.0932713	0.136711	0.17814	0.206362	0.212521	0.194573	0.158368	0.114594	0.0737159
4	0.0291877	0.0481224	0.0705344	0.0919097	0.10647	0.109648	0.100388	0.0817086	0.0591236	0.038033
3	0.0133877	0.0220726	0.0323524	0.0421568	0.0488353	0.050293	0.0460455	0.0374778	0.0271186	0.0174448

Figure 3. 4 A heatmap sample

For net training, the center point 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 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 key points in the feature map. Figure 3.4 is an example of heatmap data, Figure 3.5 is a visual example of a heatmap. That point refers to an object there because the importance is far more than others.

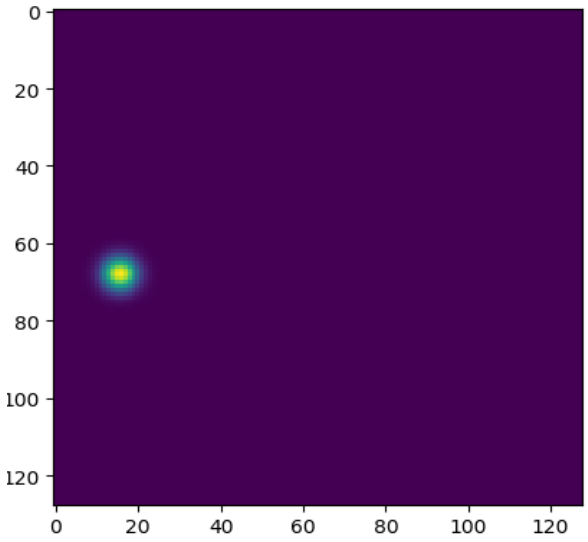

Figure 3. 5 An visual result of the heatmap

3.3.2 Loss Function

In order to train this network, a loss function is needed to be implemented. As a kind of DL-based method, it combines several parts to evaluate its performance. It is shown in eq.(3.1).

$$L_{det} = L_c + \lambda_{size} L_{size} + \lambda_{off} L_{off} \tag{3.1}$$

where L_{det} is the total loss. Equation (3.1) 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, 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.

$$L_c = -\frac{1}{N}\sum_{xyz}\begin{cases}\left(1-\widehat{Y_{xyc}}\right)^{a}\log\left(\widehat{Y_{xyc}}\right) & if\ Y_{xyc}=1\\ \left(1-Y_{xyc}\right)^{\beta}\left(\widehat{Y_{xyc}}\right)^{a}\log\left(1-\widehat{Y_{xyc}}\right) & otherwise\end{cases} \tag{3.2}$$

Equation (3.2) gives the formula on how to calculate the center loss of CenterNet, where $\alpha = 2,\ \beta = 4$. This refers to focal loss (Lin, Goyal, Girshick, He, & Dollár, 2017), which avoids weights have a dominant control and deals with the unbalance of the positive and negative samples.

As a typical problem of one-stage detectors, the unbalance of examples leads to a bad performance in the accuracy of object detection. A one-stage detector generally will generate a large number of negative samples. Only a few of those negative samples could be applied to detect objects, which causes class unbalanced problems. It will increase the loss of negative samples while they are not useful for training our model. Moreover, those samples which are easy to be detected will lead to a much tough problem, which means those easy samples will take an overwhelming scale in the training process. Each loss of those easy samples is very low, they are easy to be detected but our model could not get more information from them, the amount of those easy samples will be considerable. This disturbing problem even will make our model could not be converged.

Therefore, a focal loss is used to handle the unbalance problem. It makes the training process focusing on hard examples, instead of easy samples. In this way, it could

tremendously decrease the weight of easy negatives.

As a result, inspired by focal loss, the center loss is also designed to make this model focusing on useful information. In our model, the center points could not easily be detected to provide more loss value. The hyperparameter α, β are used to adjust the relationship between the loss of center and non-center points.

For the center loss, apart from the hyperparameter α, β. N is the number of key points of the input image, which will be used to normalize the positive focal loss to the range [0,1]. Besides, the value of Y_{xyc} could correct the training process. As a result of detection, the value of Y_{xyc} is obtained by mapping the key points of ground truth to the feature map. The value of $\widehat{Y_{xyc}}$ means that whether it is detected as a certain kind of class or not.

For instance, when training, given $\widehat{Y_{xyc}} = 1$, that means it is a point that is easy to be detected as a class of visual objects. For this point, $\left(1 - \widehat{Y_{xyc}}\right)^{a}$ will be used to minimize the L_c due to we need more other information from different parts, instead of this part, to train the net better. It is a method to avoid easy samples and make the training focus on the hard sample.

In order to avoid overfitting, L_c will be decreased by this value of $\widehat{Y_{xyc}}$. It means that the CenterNet has had a good performance at this point. 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 due to these features will have excessive weight. Hence, this model needs to use $\left(1 - \widehat{Y_{xyc}}\right)^{a}$ to decrease the contribution of their weights.

On the contrary, when training, given $Y_{xyc} = 1$, but with $\widehat{Y_{xyc}}$ is close to 0. That means our model could not detect this object correctly. This model has not learned information from this point. So, $\left(1 - \widehat{Y_{xyc}}\right)^{a}$ will be used to increase the L_c to make our model learn more information from this object. In this way, this model could be trained better through the feature from this center point.

With the center point, equation (3.2) deals with the weight of the neighbor of the center point of the true center point. Oppositely, Y_{xyc} will not be 1, given 0. Now that, the value of $\widehat{Y_{xyc}}$ should be 0. However, if it is not 0 or even close to 1, which means our model incorrectly detects this object, this model needs to penalize this negative sample. To make it come true, $\left(\widehat{Y_{xyc}}\right)^{a}$ will get increased in this condition, which will make this hard object have a bigger influence on the L_c to accelarete the converge speed.

For those points which are closer to the center point, the values will be very near to 1. However, they will influence the real center point. For them, because the distance will be increased from far to near, the item $\left(1 - Y_{xyc}\right)^{\beta}$ will be decreased from far to near. So, the influence of neighbors of the center point will be limited. In this way, the learning will be concentrated on the center of the feature map, which is the most important feature in the feature map.

It is a very important idea of how this model to learn without the anchor. To achieve this purpose, our model will only focus on the center of each object, just list what the anchor method doing. So, we need to increase the weight of the center point while reducing the weight of its neighbors. Hence, we will combine the two parts of values depending on the distance between them and the center point. For those points which are near to the center point. Given a truth value Y_{xyc} as 0.9, which nearly equals 1, while the predicted value $\widehat{Y_{xyc}}$ is also near 1, which is not correct because they are not the real center, so they should be limited to near 0. In order to constrict their usage, they will be punished by $\left(\widehat{Y_{xyc}}\right)^{a}$ to increase the weight. Meanwhile, $\left(1 - Y_{xyc}\right)^{\beta}$ is used to adjust the loss smoothly. So, these two parts of loss are combined to limit the total loss of the points which are closer to but not the real center point.

Of course, they are also used to deal with the points which are far from the center point. Similarly, to the value of those points, given a truth value Y_{xyc} as 0.1, which nearly equals 0. But, the predicted value $\widehat{Y_{xyc}}$ is near 1, for example, 0.9. It is not correct, because it is far from the center point, the value should close to 0. Thus, it should be

punished by using $\left(\widehat{Y_{xyc}}\right)^{a}$ to increase the loss proportion of this point in all center loss. Meanwhile, if the predicted value $\widehat{Y_{xyc}}$ is near to 0, in this correct condition, $\left(1 - Y_{xyc}\right)^{\beta}$ will strengthen the weight of the points which are far from the center point. This method will make our model can learn the critical features from those points which are far from the center. In this way, it factually enhances the center point while undermining the neighbor points. It is a useful method to tackle the unbalance of positive and negative samples.

In this model, each object corresponds to a center point, which avoids the process of handling the prior box, compared with Faster R-CNN and SSD. Through this, it could save the computational time which was cost on processing the anchors. This model also decreases the negative center pointer by using the idea from focal loss.

In a word, inspired by focal loss, this center loss could be evaluated by combining the scores between the center point and the adjacent points. It could adaptively check the easy and hard samples, to make our model more focus on the hard sample to avoid using the anchor. As the result, it could simplify the network structure and make sure accuracy, while decreasing the calculation.

After trained our model by using the center loss function, the offset loss for each center point will be performed. The input images are downsampled by a rate of 4. So, when 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 used to evaluate it.

$$L_{off} = \frac{1}{N}\sum_{p} \left|\widehat{O_{\tilde{p}}} - \left(\frac{p}{R} - \tilde{p}\right)\right| \tag{3.3}$$

Equation (3.3) shows the offset loss function, $\widehat{O_{\tilde{p}}}$ is the predicted offset. This offset loss is used to calculate the average offset by using L_1 norm loss method. In this equation, p is the input image, R is the downsampling rate. For our model, an image with a size [512, 512] will be downsampled to [128, 128]. It will also be upsampled from [128, 128] to [512, 512]. Among this process, the location of the center point will have a precision loss, which is called offset loss. In order 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 their absolute value will

be averaged by using the L1loss method.

Now that, the loss of location and offset of center point could be used to train our model. Unlike the previous anchor-based model, CenterNet is a kind of anchor-free method. Therefore, it has not a prior box. With the same aim, to evaluate the size of each object, it will predict a size for each object. It also uses an L_1 norm loss function to simplify the calculation, similarly with offset loss, which compares the predicted value and truth values.

$$L_{size} = \frac{1}{N}\sum_{k=1}^{N}\left|\widehat{S_{p_k}} - S_k\right| \tag{3.4}$$

Equation (3.4) shows the size loss of our model. Inside, $\widehat{S_{p_k}}$ is the size that our model predicts. $S_k = (x_2^{(k)} - x_1^{(k)}, y_2^{(k)} - y_1^{(k)})$ 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 this L_1 norm method. As a result, the size loss will be obtained.

Through the above processing, we could know that the CenterNet combines three different loss values: Center loss, size loss, and offset loss to evaluate the performance of this model. The CenterNet uses the center loss method, which is inspired by the focal loss method to handle the hard easy sample problem. Hence, it could make this model adaptively learn those samples and features with higher weight. As a result, it could make this model converge earlier. Besides, it also uses the L_1 loss method to assess the size loss and the offset loss to confirm the accuracy and simplify the calculation.

3.3.3 Backbone Module

We have introduced how to evaluate and train our model based on its loss function. We now detail how to obtain the feature map from raw images to generate the heat map. For our proposed deep learning model, we choose three methods as its backbone module to extract future maps. Due to the limit of our hardware, the lightweight methods were considered preferentially. We mainly take into account ResNet-18, DLA-34, and Hourglass as our backbone networks. Then we will implement them by using our dataset

to observe the results to evaluate which one is suitable for our project.

For the CNN network, its performance 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, owing to its shut cut structure (He, Zhang, Ren, & Sun, 2016). It adds a shortcut x in this structure, this structure could simply the training process and settle the degeneration problem. Meanwhile, an 18-layer residual network is chosen as the backbone of our model. The second backbone of our model is DLA (Yu, Wang, Shelhamer, & Darrell, 2018), which merges the feature from different depths.

Figure 3. 6 The structure of DLA

In Figure 3.6, 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 could improve the utilization ratio of features between different layers or blocks.

The last method is the Hourglass network (Newell, Yang, & Deng, 2016). The structure is shown in Figure 3.9. The original purpose of Hourglass was used to catch the information on each scale to avoid missing small local features. Hence, its structure is bilateral like an hourglass. On the left part, from c1 to c4, 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 $c1a$ to

$c41$.

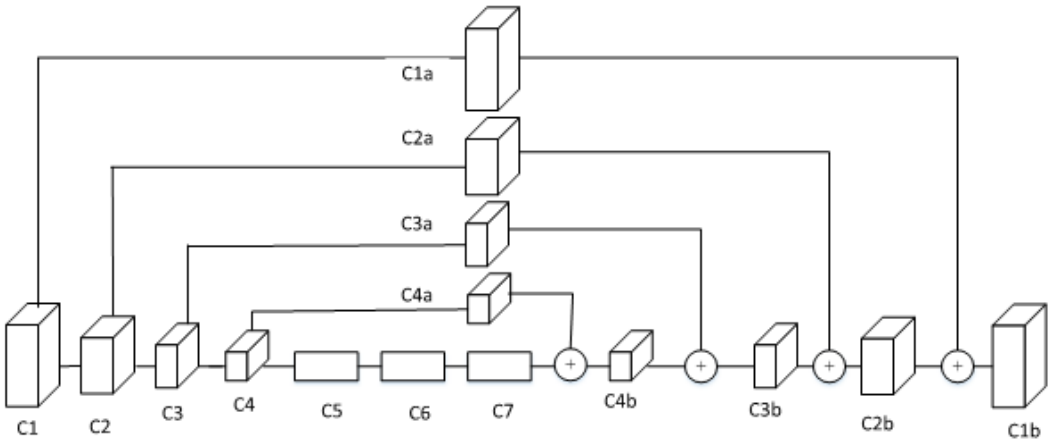

Figure 3. 7 The structure of Hourglass

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 $c1b$. Hence, Hourglass utilizes the hourglass model to extract as many features as it can. It was used as our third backbone model.

3.4 Model Evaluations

For the result of our model, it is hard to achieve a fast speed and high accuracy with small memory. We have to evaluate them separately. There are several metrics to evaluate the performance of our models, such as precision, recall, and accuracy.

Firstly, accuracy tells us how many true samples were selected from all samples, which could give us a standard to evaluate a model. Equation (3.5) shows us how to calculate accuracy.

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+TN+FN} = \frac{TP+TN}{All\ detections} \tag{3.5}$$

where TP is the number of true positive samples, which means that this sample is positive and we correctly classify it, TN is the number of true negative samples, which refers to that this sample is negative, we still classify it correctly. FP is false positive, it is the number of those that should be positive samples, but we classify it as a negative sample

incorrectly. FN is a false negative, it is the number of those actually in the negative sample, we incorrectly classify it as the positive sample. Equation (3.6) shows us how to calculate the precision.

$$\text{Precision} = \frac{TP}{FP+TP} = \frac{TP}{All\ positives} \tag{3.6}$$

Precession was calculated by using TP to be divided by the sum of FP and TP. The numerator is the number of the positive sample we correctly classify, the denominator is the total number of all positive results which including we classify correctly and incorrectly. Hence, precision is thought to evaluate the ability to find the correct positive sample from all positive samples.

$$\text{Recall} = \frac{TP}{FN+TP} = \frac{TP}{All\ ground\ truths} \tag{3.7}$$

The recall was calculated by using equation (3.7), the denominator is the sum of FN and TP. FN is the sample should be positive, but incorrectly classified as negative. Hence, the sum of FN and TP could be thought of as all ground truths. Hence, the recall evaluates the performance of our model about its ability to detect those positive samples from all positive samples.

$$mAP = \frac{\sum_{q=1}^{N} AveP(q)}{N} \tag{3.8}$$

Besides accuracy, precision, and recall, another to assess the performance of object detection is mAP as shown in equation (3.8), *N* is the number of classes in this model. AveP(q) is the area related to the precision-recall curve (PRC).

Chapter 4
Results

For this chapter, we will demonstrate the results of our proposed model by using our own dataset. The structure of our model mainly is CenterNet, an anchor-free DL-based method. We apply three different types of backbone modules for the proposed CenterNet. Hence, the performances based on our dataset will be compared. The convergences will be shown by using the loss curve. The backbone networks that we adopt include ResNet-18, DLA-34, and Hourglass.

4.1 The Result of CenterNet

Figure 4.1 shows us an example of the result. We find that there are four fruits inside this image. All fruits are labeled by using a bounding box with different colors, as well as a class label and confidence value.

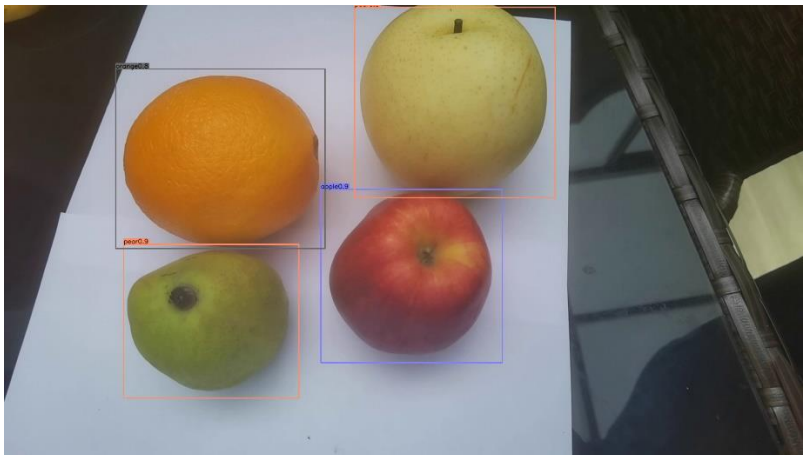

Figure 4. 1 The sample of our detection results by using CenterNet

4.1.1 The Performance of Convergency

From our experiment, the performance of CenterNet with different backbone nets is desirable by using our dataset, we make use of the loss function to observe the convergence of model training.

The loss function of CenterNet includes three parts, which consists of heatmap loss also named center loss above, offset loss, and size loss. The heatmap loss is named hm_loss with orange color. The size loss is marked as wh_loss with green color, this takes its width and height into account. The offset loss is denoted as off_loss with red color. Finally, all the losses are combined as a representative loss.

In Figure 4.2, we see that our model based on DLA-34 converges rapidly. Even if there are just twenty epochs, it could converge very fast.

The convergency speed of the model with ResNet is not as fast as DLA, especially wh_loss has an obvious decrease at the 20-th epoch. Finally, its loss remains stable at the 60-the epoch. Actually, for this model, there should be more epochs, but the final result

is acceptable.

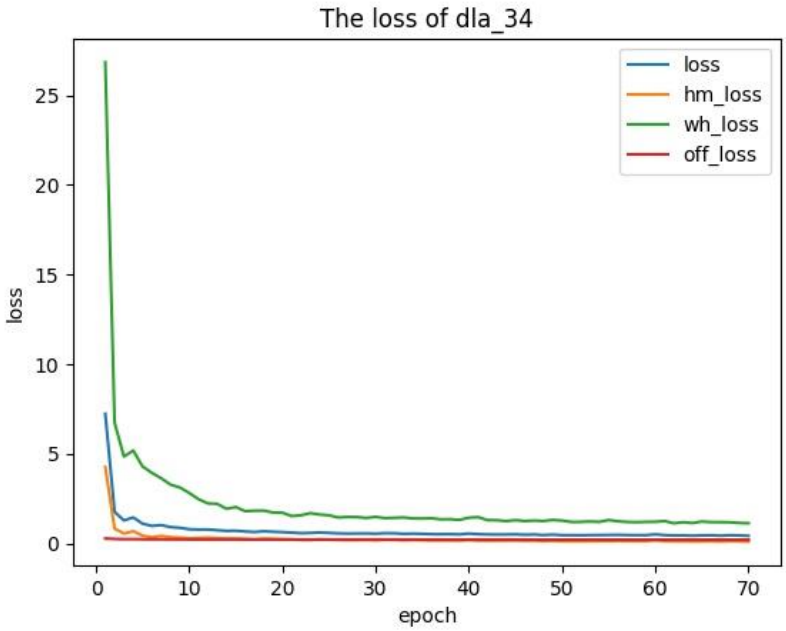

Figure 4. 2 The loss performance of the model DLA-34

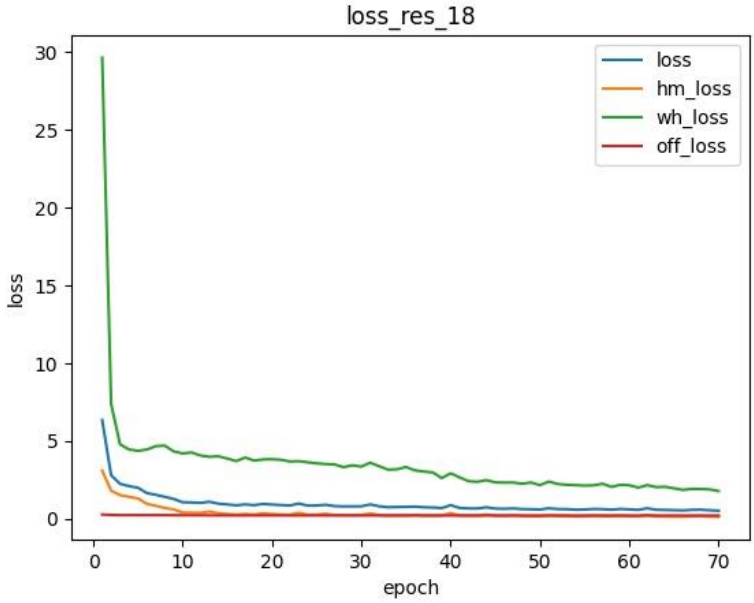

Figure 4. 3 The loss performance of the model ResNet-18

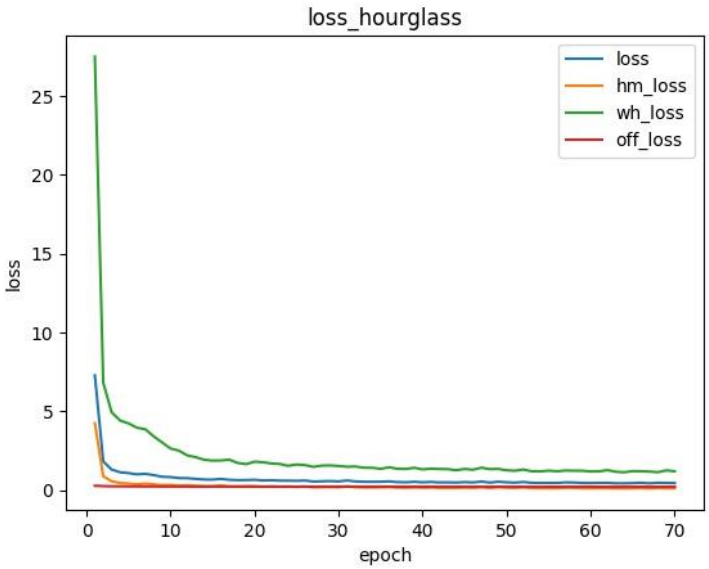

Figure 4. 4 The loss performance of the model Hourglass

In Figure 4.4, we see the shape of our loss of hourglass is very smooth. It could be converged at the 20-th epoch, and have the same convergency performance as the first one. We also see that all the models have good convergency in our dataset. Even though the model ResNet-18 is not as fast as the others, but the final performance is as same as theirs.

Both DLA-34 and Hourglass have a good performance at the speed of convergences. From observing their structure, we find that the ResNet-18 does have a wide process path to connect different branches. Even if it is the first network to use the shortcut to connect different depth convolutional layers. As the consequent network, DLA-34 not only has the IDA to connect the different layers to transform the feature information from bottom to above like what the ResNet does, but also merges the feature from different branches and scales. In this way, it could enhance immensely the ability to combine different features to reduce redundant computation. The structure of Hourglass shows that it also has this advantage, which could make sure the speed of its convergency.

In a word, all of these three models can converge at a limited time due to our proposed model based on a focal loss function. DLA-34 and Hourglass have a better performance than ResNet-18.

4.1.2 The Time Costs

After comparing the convergency of our models based on different backbones. The time cost of each model is listed in Table 4.1.

Table 4. 1 The time cost of each model (second)

	Loading	Preprocessing	Object Detecting	Post-processing	Totals
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

From Table 4.1, we find that the model has the most little time cost which is based on ResNet-18. The total time is only 0.028s, which is nearly half of DLA-34 and Hourglass. For DLA-34 and Hourglass, their time cost is very close. The total time for DLA-34 is 0.055 second, and 0.056 second for the Hourglass model.

All these three models have a similar distribution of time cost. The time spent on loading, preprocessing and postprocessing does not reach half of the whole time. The most consuming part is 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.

For the reason why the speed of ResNet-18 is the fastest one, it is also revealed in the designed structures. From their structures, we find both DLA-34 and Hourglass have a much complex structure rather than ResNet-18. This is why these two models have a fast speed of convergency.

4.1.3 The Accuracy Analysis

After discussing the convergency and time cost, we analyze the convergence and time cost for each model, we will discuss the accuracy of each model.

Table 4. 2 The average accuracy of the model DLA-34

DLA-34	Precisions	Recalls
1	0.870	0.744
2	0.995	0.898
3	0.987	0.899
4	0.801	0.845
5	0.878	0.906
mean	0.906	0.858
mAP	0.9	

From Table 4.2, we see that the best precision of DLA-34 is 0.995, meanwhile, accompanied by 0.898 of recall. The lowest score of precision is 0.801 with a recall of 0.845. The mean precision is 0.906, and the mean recall is 0.858. Finally, the mAP of this model is 0.9.

Table 4.3 shows that the best precision of ResNet-18 is 0.953, but the average precision is 0.8194. On the other hand, the best recall is 0.827, along with an average recall of 0.765. The mAP value of ResNet is 0.786, which is lower than the DLA-34. The performance of DLA-34 is better than ResNet in the aspect of accuracy.

Table 4. 3 The average accuracy of the model ResNet-18

ResNet-18	Precision	Recall
1	0.781	0.665
2	0.953	0.816
3	0.914	0.818
4	0.655	0.697
5	0.793	0.827
mean	0.8194	0.765
mAP	0.786	

Table 4. 4 The average accuracy of the model Hourglass

Hourglass	Precision	Recall
1	0.845	0.722
2	0.996	0.878
3	0.975	0.877
4	0.717	0.765
5	0.855	0.887
mean	0.878	0.826
mAP	0.88	

The best precision of Hourglass is 0.996 and the average precision is 0.878. The best recall is 0.887 and the mean recall is 0.826. Both precision and recall are higher than ResNet-18. But they are not better than the result of DLA-34.

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

4.2 A Summary

In this chapter, we present the data on the convergence performance of the loss function for each model. As well, we compare all of their time cost and the accuracy based on the mAP. We find that the results converge rapidly in a limited and acceptable time. Hence, all of these models could be used to tackle object detection with our own dataset. Moreover, from the data of time cost, ResNet-18 is thought of as the fastest model for object detection, its training speed is two times faster than DLA-43 and Hourglass. The DLA-43 and Hourglass have similar time costs. Finally, the accuracy of the three models was given by comparing precision, recall, and mAP. The DLA-43 model has the best result with 0.9 than 0.786 for ResNet-18 and 0.88 for Hourglass.

Chapter 5
Analysis and Discussions

In this chapter, experimental results are analyzed and compared. Our comparisons of the experimental results under various conditions will be detailed. Finally, we give a summary based on our experimental results. DLA-34 is the most suitable method for our model, based on the performance of speed and accuracy of model training. The limitation is given at the end of this chapter as well.

5.1 Analysis

In our project, three CenterNet models based on DLA-43, ResNet-18, and Hourglass were deployed to handle the fruit detection on our dataset. In the following sections, we will discuss the three models based on our results.

After the analysis, we compare the convergence performance, time cost, and accuracy of the three models. From the results, we see that all of them have a similar convergence performance. All of them converge at the 50-th epoch, which means those backbones could be trained from the feature of our dataset. Hence, these models are suitable for our dataset. Especially, both DLA-34 and Hourglass converge very fast.

The reason why DLA-34 and Hourglass converge very fast is the well-designed network structure. Just as in Figure 3.6 and 3.7, both of them have a more complex structure to deal with the information which got from different branches, not only limited by the depth.

As the first net to put forward the shortcut method, the ResNet block can transmit the feature information from the different convolutional layers in the network. But, it is limited by the shortcomings that it can not combine the feature information from different branches, or it is limited in width.

DLA-34 uses the aggregation module to combine the feature from different depth layers rather than the method of ResNet which just transport those features to more deep layers. Besides, DLA-34 also uses the HDA module to combine the feature from different scales by upsampling. More specifically, DLA-34 can transmit the information from bottom HDA modules to top HDA modules. In this way, it reuses and integrates these feature information from different layers, just like the method of ResNet.

As a typical encoder to decoder structure, Hourglass merges the information from multiple scales like DLA-34. The purpose is to catch all the features in various scales from an image, which makes use of downsampling and upsampling to avoid losing the small object. In the downsampling part, it will employ the pipeline to deal with the input image in multiple scales. Then, in the upsampling part, it will merge those features extracted from the downsampling part. Hence, it can also efficiently merge and reuse the

feature information from the input image, which is similar to the DLA-34. In a word, DLA-34 and Hourglass have a better performance in the convergency than the ResNet-18.

However, for time-cost analysis, the model ResNet-18 has the lowest time cost with just 0.028 seconds in total, the costs of DLA-34 and Hourglass are very similar, which are two times slower than ResNet-18. This speed is still acceptable for our research project. However, for their accuracy analysis, the performance of ResNet-18 is not as good as in the time cost analysis, which is related to the speed. The best performance of accuracy among those three models is from DLA-34.

After the analysis of the performance in convergency, we know the structures of DLA-34 and Hourglass are more complex than the ResNet. Through those complex structures, DLA-34 and Hourglass could have better results of accuracy than ResNet. Those complex structures, for example, HDA and IDA of DLA-34, downsampling, and upsampling for Hourglass, could help DLA-34 and Hourglass to extract more useful and complicated feature information. Hence, DLA-34 and Hourglass could have a better performance on accuracy.

However, these advantages will limit the speed of their performance in time cost. In other words, the reason why ResNet-18 has better performance is due to its simple structure. Unlike the complex structure of DLA-34 and Hourglass, ResNet-18 need not deal with the information acquired from different branches. It just passes those feature maps from the layer in different depths. This condition helps it to save a lot of computational time.

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

Figure 5.1 shows the image samples for CenterNet based on DLA-34 net. From Figure 5.1, we find that those full-view fruits should be detected from an image perfectly. Even if there are several fruits at the corner being ignored by CenterNet but for those central fruits, our model can find them easily.

Figure 5. 1 The successful examples for fruit detection based on DLA-34

Figure 5. 2 The missed fruits detected by using CenterNet based on DLA-34

Figure 5.1 and Figure 5.2 show us a view of the results of CenterNet. We find that the results are quite well. As shown in Figure 5.1, each fruit is detected by using our model very clearly. For the second one, even if some fruits are not correctly detected, the results are still acceptable.

Figure 5. 3 An example of single apple detection

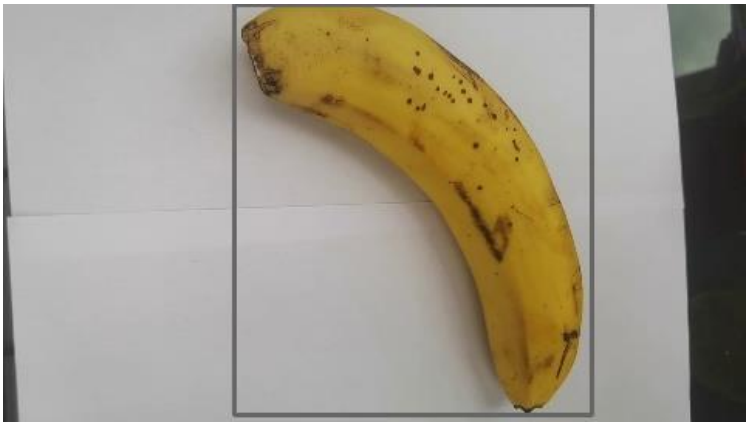

Figure 5. 4 An example of single banana detection in digital image

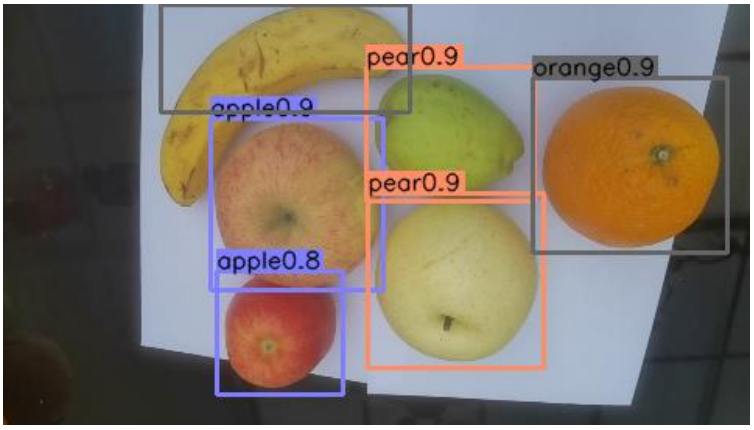

Figure 5. 5 An example of multiobject detection from a given image

Figure 5.3 and Figure 5.4 show examples of single object detection from an image. Figure 5.5 demonstrates multiobject detection from a given image, which reveals that our model is able to detect both single object and multiobject with high accuracy. Besides, the size of this apple has a considerable difference in Figure 5.3 and Figure 5.5. Our model still is able to detect them from the given images.

In this thesis, we find the performance of our network is quite well. It cannot only detect the class of our fruit correctly but also mark those fruits appropriately. It means all three heads, which are center, size, offset, work well.

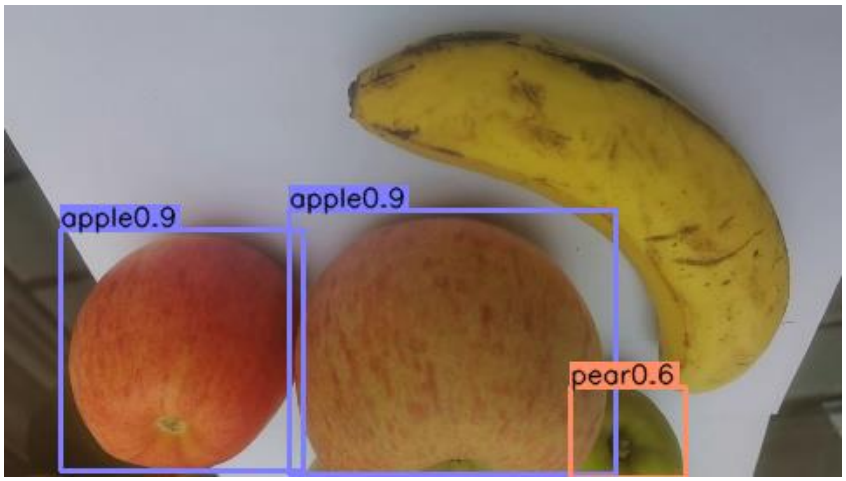

Figure 5. 6 An example of missed detection

Figure 5. 7 An example of missed detection due to overlapping

Figure 5.6 and Figure 5.7 show us two failed cases of fruit detection. In Figure 5.6, it does not detect this conspicuous banana, but it does detect the pear in the corner with a

confidence of 0.6, The pear is hard to be found, even by using our human eyes. Figure 5.7 shows that there are two pears at the bottom-left corner of the image, but our model could not detect them. Figure 5.8 shows that a pear is occluded by apples and pears. Our model is still able to detect the green pear with a confidence of 0.4.

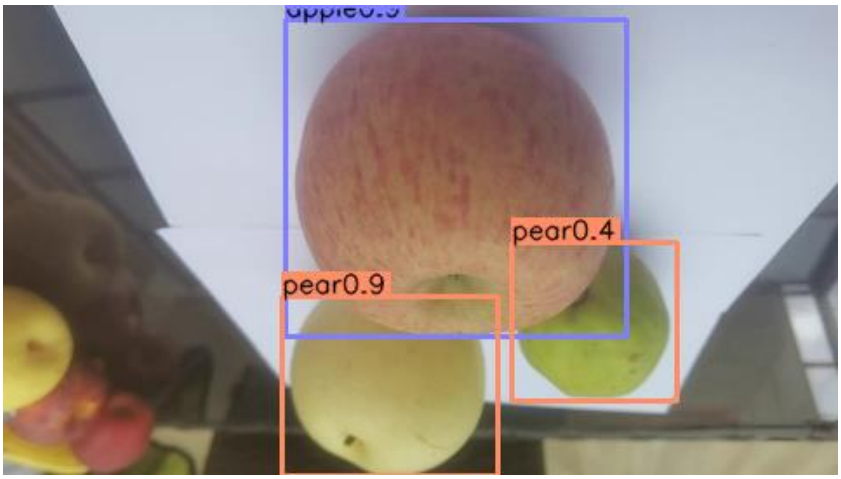

Figure 5. 8 An example of correct detection due to overlapping

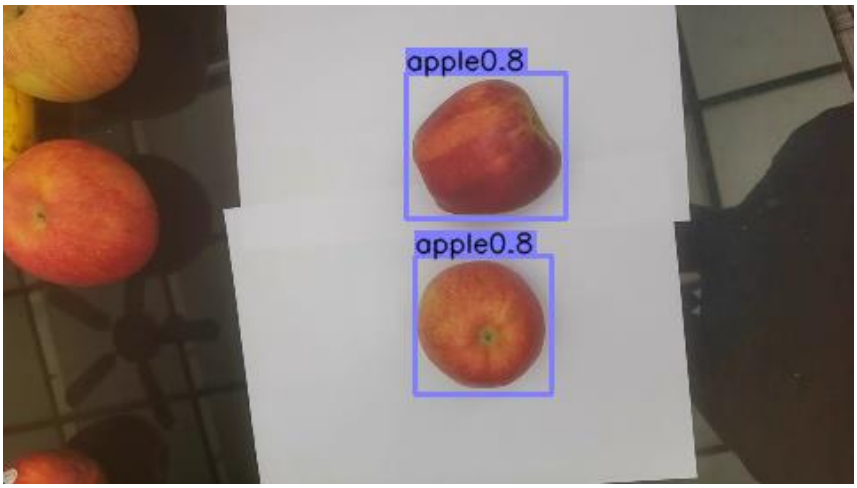

Figure 5. 9 An example of missed detection

From Figure 5.9, we see that there are several apples on the top-left missed, even if they are very obvious for our humans. In Figure 5.9, we see that our model based on DLA-34 is available for fruit detection, though there are still problems. For these problems, the following part will discuss the potential reason why our model loses those objects.

CenterNet, as a kind of anchor-free detector, finds the center point of an object from a heat map instead of from anchors. As the methodology part description, the heat map is mapped from the feature map by using a Gaussian kernel. Then, we use a kind of focal loss function to train our model.

Among these steps, if two objects are very closed to each other, their center point will influence each other. Generally, there two objects will be thought of as one object after the downsampling. The reason is if they are very closed to each other in the original image, after the downsampling of 4, the distance between them will be squeezed strongly, then they will be thought of as one object in the heat map. Due to the same reason, if the center points of two objects close to each other after downsampling, our model will only detect the center point with the bigger value. An example is shown in Figure 5.10.

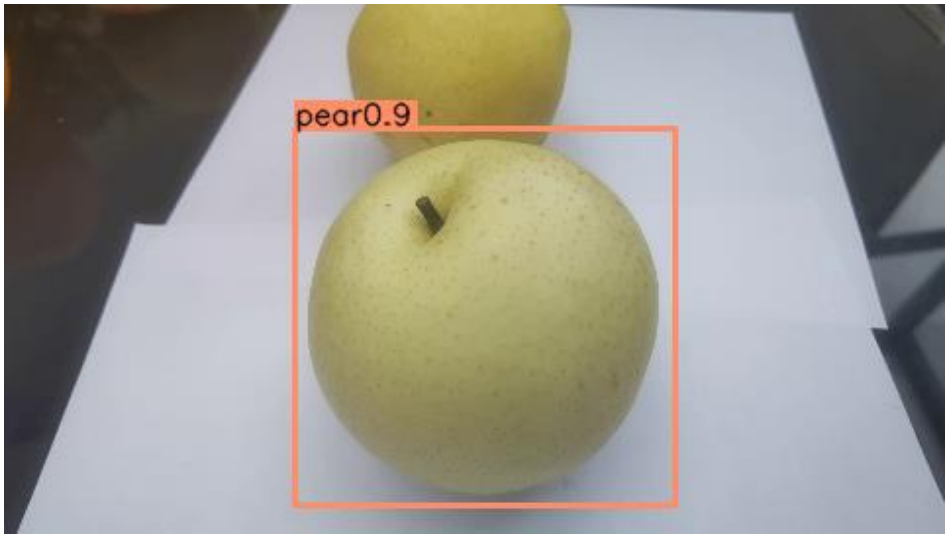

Figure 5. 10 An example of occlusion

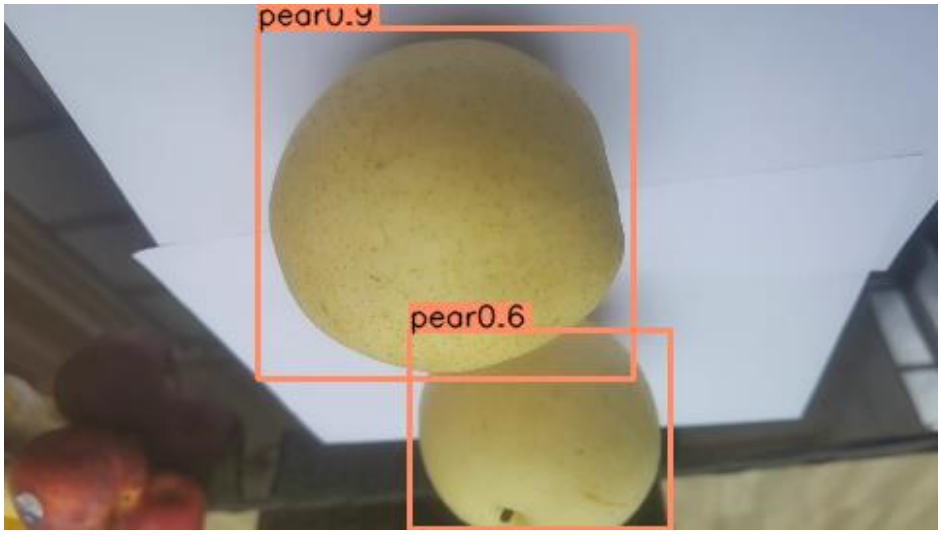

Figure 5. 11 An example for correctly detect occlusion

The occlusion problem is not always happened, for instance, in Figure 5.11, our model successfully detects this overlapping problem, even though they have the same class. So, considering the accuracy of our model based on DLA-34, we could accept the casual mistakes.

As a kind of one-stage detector without anchor, its performance on our dataset is quite good. The accuracy of the anchor-free detector is rather high based on our dataset by using three heads from the heat map. As a one-stage detector, the speed is also very impressive though its detection methods are based on feat maps without complex and redundant computation compared to SSD and Faster R-CNN.

Hence, considered the performances of speed and accuracy, this CenterNet model based on DLA-34 is acceptable for our fruit detection. As a kind of DL-based methods, it combines the development of one-stage and anchor-free methods to provide a tradeoff between speed and accuracy.

5.2 Discussions

In this session, in order to avoid the disadvantage of the two-stage detect method which will lead to a slow speed, as well as of the anchor method which will produce a redundant computation, the CenterNet network is utilized to detect fruits by using our model. By using the center point of a visual object from the heat map, it could achieve the ideal

result.

In our experiments, three DL-based methods were used to be the backbone of the CenterNet model. All the methods could converge after 50 epochs, ResNet-18 is a little bit faster than Hourglass and DLA-34, even though, the speed of Hourglass and DLA-34 is still acceptable. After accuracy analysis, we find that the DLA-34 has the best result based on mAP.

For the results of convergency, speed, and accuracy, we compare the differences between their structures. As the backbone of our model, the ability of each network is compared based on this same dataset.

As the first method which was put forward to use the shortcut, ResNet-18 is successful. However, limited by the structure itself, it can only transmit features from different convolutional layers, even if it can jump over several layers, but it can not merge or combine those features. Hence, this shortcoming limits its upper bound in accuracy and converge speed. So, its performance on accuracy is not as good as on speed.

Hourglass overcomes the limitation of combing different features to make the detection process more efficient by upsampling to merge different features. It uses downsampling to deal with the input image to extract features in different scales. In this way, each convolutional block is able to deal with different scale features in a very subtle scope. Following, upsampling is used to combine those features from different scales to implement the most detailed feature consolidation. As the result, its speed is a little slower than ResNet, but the accuracy is quite good.

DLA-43, as a consequent model of GoogLeNet, combines both the features from the depth and the width, which uses the aggregation module to combine the small feature in the first step. It uses the HDA module to merge those features extracted by using aggregation modules. After these steps, it fuses those features extracted from the HDA module with different depths. Finally, all the feature information from different depths and widths is used to detect an object by merging them from different modules.

As the experimental results show, the proposed model achieves the highest score in the aspect of accuracy, which is much higher than ResNet-18. For the aspect of speed, even if it is similar to Hourglass, and slower than ResNet-18, considering the balance of

speed and accuracy, we think this model based on DLA-34 is acceptable.

From the final image data, we find that the CenterNet based on DLA-43 is able to detect fruits from an image. For those images, our CenterNet could detect a single fruit or multiple fruits together. For those occluded fruits, our model still detects the fruits which are overlapped. Thus, this model is also able to detect fruits from digital images.

5.3 Limitation

As a kind of DL-based method to handle this fruit detection problem, our proposed model CenterNet based on DLA-34 can fix it very successfully, which has a good performance based on both speed and accuracy.

However, though this model has a fairly good performance base on our dataset, there are still several problems. Firstly, the structure of our model, as a tradeoff between speed and accuracy, can be improved. A newer method can be deployed to compare with our model.

As the inherent problem of the heat map, it will miss small objects if two centers of these small objects are too near. Especially, after downsampled four times, the small objects in the same class will be thought of as a part of the big object, which will lead to the overlapping problem.

With a good ability to detect fruit, our model is thought of as a good DL-based method. However, DL-based methods generally need more data to be trained so as to achieve better performance. In our research work, limited by the time and the resource, we can not obtain so many images. Hence, we only detect four classes of fruits with 1,600 images or so. Meanwhile, the sample diversity is also limited by our dataset. Hence, in the following sections, more datasets and classes can be collected for the models.

Even though our model has a quite well performance in this research work. However, just as we stated in our methodology, our model is simplified to make it suitable for our developing platform. Limited by GPU memory, the ResNet-18 is deployed rather than more powerful methods such as ResNet-101. With a more powerful platform, the ResNet-based methods will acquire a much better performance than this simplest ResNet-18. Of

course, a more powerful platform is much useful for other complex models. Especially, to train different models and to compare their performances, an ideal platform will lead to a better result, typically, in the speed and accuracy of the desirable models.

Briefly, even if with good performance, our research work still has several limits. Mainly, the developing platform limits the structure of our model, which leads to that only the small and simple network was possible to be implemented. Limited by the structure of CenterNet, it has some limitations whilst processing the small objects which are squeezed together. As a DL-based method, more datasets could aid us to get more features so as to achieve better performance.

Chapter 6
Conclusion and FutureWork

In this chapter, we will summarize our methods of this project, including how we start our research and the background of object detection based on two branches, ML-based and DL-based methods. We also talk about our model based on three different backbones. Finally, we showcase the outcomes of this research project. The limitations and future work will be analyzed at last.

6.1 Conclusion

The purpose of this thesis is to find a practical model that could settle fruit detection from digital images. For this purpose, our methods from two main branches of computer vision were discussed. Firstly, three models are proposed based on ML-based methods, which comply with the same process, region selection, feature extraction, and classification.

After reviewed ML-based methods, in this thesis, we overview the DL-based methods. Thus, in this thesis, we firstly introduce the history of ANNs. Then, the development of ANNs also is detailed from Perceptron, MLP, DBN, CNN, LeNet, AlexNet, VGGNet, GoogLeNet. After listing the achievement of CNN in image classification, we delineate the development of object detection based on the DL-based methods. R-CNN, as the base of the DL-based method, has been employed for visual object detection in computer vision for several years. As the successive new model, it is influenced more or less, which keeps the working process the same as the ML-based method but changes the region selection method from the sliding window method into a selective search method.

Moreover, the most important change it makes is that it deploys the CNN network to extract features rather than using the traditional ML-based feature extraction method. Followed R-CNN, Fast R-CNN and Faster R-CNN were designed for object detection questions. This thesis also briefs YOLO and SSD methods. Finally, the structure of CenterNet is expounded with three backbone: DLA-34, ResNet-18, and Hourglass. In the last part, we compare the performance of the model based on various backbone modules.

In this chapter, the difference in the structure among these three backbones is identified. ResNet-18 as the simplest model achieves the fastest speed. However, its performance of accuracy is not ideal as the accuracy is the lowest. The reason is thought of as that its structure only transmits the information between various depths, and does not merge them. In this way, it could save time, but it will also limit its ability for extracting and utilizing those features.

After fully analyzed ResNet-18, in this thesis, the performances based on the backbones of Hourglass and DLA-34 have been stated. The hourglass was designed to

extract features from an image, its structure looks like an hourglass, consisting of two main parts, downsampling and upsampling. The downsampling is applied to extract features as more as it can get from the image, upsampling is employed to merge those useful features. Through that method, it can detect various objects of different sizes. Hence, it has a better performance on accuracy than the ResNet-18. Even if its speed is not faster than ResNet-18, but still acceptable.

With a distinctive structure, DLA-34 is introduced. From the structure of DLA-34, we know that its idea is similar to Hourglass, which combines and merges different features not only from different depths but also from different widths. But its structure is more complex than Hourglass, hence, it extracts feature information and merges them in a different way. It combines the feature from several aggregation modules by using HDA modules. Those features from different scales and widths will be merged to be some more complex features just like what our visual cortex doing. Then, IDA modules will be used to combine these features from different depths. As a result, it can combine the feature information from different depths and widths, meanwhile, the postprocessing can assist this model to be trained better because of those more useful features.

Eventually, DLA-34 was chosen as the backbone to design CenterNet. For training this CenterNet model, the loss function is needed to evaluate the performance of our model. It consists of three parts. Based on the original idea, it needs to consider the center point loss, the offset of the center point, and the size of the predicted box. Hence, this thesis combined these three losses to assess our model.

Moreover, to overcome the disadvantage of one-stage about easy sample dominant problem, a focal loss-like loss function is used in this thesis. The reason why this problem will undermine our detector is that even each negative sample loss is very small, with a large number of them, the amount will be very large. It will make our model hard to be converged.

The loss function inspired by the focal loss method will make the training of our model concentrate on hard samples, instead of easy samples. This will accelerate our training process by adaptively selecting the hard samples.

Throughout these methods, our model could obtain a tradeoff between speed and

accuracy. After model training and testing, the results show that this model could be used to tackle the fruit detection problem by taking account of both speed and accuracy.

However, there are still several problems with this model, which mainly are the missed detection, limitation about hardware or software. Firstly, if the center points of two objects are too closed after downsampling, it casually leads to the missed objects in the detection, especially for those small and squeezed objects. Then, limited by the hardware of our developing platform, especially for the memory of GPU, our backbone net is simplified to fit our platform. This change limits the ability of our model and impairs the performance of our model, which could not present the best performance based on a more complex structure. As a typical DL-based method, more data will help our model get better performance. Limited by the time and resources, our dataset is also not sufficient in quantity and classes.

All in all, in this thesis, we put forward fruit detection in the agricultural harvest robot. 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, such as VJ-method, HOG+SVM, and DPM methods. Then, we reviewed the DL-based methods. Inspired by human brain structure, the DL-based methods have a great development in the last decades, which has overpass the ML-based method and our human visual system. Hence, we shortly introduced the background and history of ANNs based on classification, from Perceptron to MLP, LeNet, AlexNet, VGG, GoogLeNet. Then, we discuss the achievement of CNN in object detection, from the base R-CNN, Fast R-CNN, Faster R-CNN, YOLO, SSD to CenterNet. We concisely present the innovation and shortcoming, finally, CenterNet is decided to be used as our proposed model.

After designing 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 result of our model, considered the convergency, speed, and accuracy, the DLA-34 method is finally chosen as our backbone to tackle this fruit detection problem.

Compared with those three backbones, we give the limitation of our research.

Limited by the inherent of our model, the process of mapping the input image with the feature map will lead to a casually missed detection. Limited by the developing platform, the ability of our model is undermined to make it suitable for our GPU. We also believe more data is useful to improve the performance of DL-based methods. In a nutshell, the CenterNet based on DLA-34 is a method that is able to successfully handle our fruit detection problem.

6.2 Future Work

In this thesis, our dataset maybe not enough. 1,690 images were collected in our dataset, 1,352 of them were used for training. Hence, only less than 400 images for each class. For the DL-based methods, more training data means the network could be trained much better for the reason that those data contains a vast deal of useful noises. Those noises could help DL-based methods to alleviate overfitting.

For the backbone nets, the reason why we chose DLA-34 is that our hardware is limited in CPU and GPU. Hence, we could not design a huge model, otherwise, we would not deploy and train it based on our laptop computer. In future, a more powerful backbone and training platform should be accommodated to design a better model. This will help us to design and compare those models more quickly and easily. The more important issue is that it could make sure we will test our best model based on the limited dataset.

For the fixed problem of CenterNet, in future work, the new tricks are employed to deal with those little problems. The CenterNet is directly replaced by a newer model to achieve a better result for this fruit objection problem.

References

Amari, S. I. (1993). Backpropagation and stochastic gradient descent method. Neurocomputing, 5(4-5), 185-196.

Agrawal, S., & Khatri, P. (2015). Facial expression detection techniques: Based on Viola and Jones algorithm and principal component analysis. In International Conference on Advanced Computing & Communication Technologies (pp. 108-112).

Arcaro, M. J., & Livingstone, M. S. (2017). A hierarchical, retinotopic proto organization of the primate visual system at birth. *Elfie*, *6*, e26196.

Azevedo, F. A., Carvalho, L. R., Grinberg, L. T., Farfel, J. M., Ferretti, R. E., Leite, R. E., ... & Herculano-Houzel, S. (2009). Equal numbers of neuronal and nonneuronal cells make the human brain an isometrically scaled-up primate brain. *Journal of Comparative Neurology*, *513*(5), 532-541.

Bargoti, S., & Underwood, J. (2017). Deep fruit detection in orchards. In IEEE International Conference on Robotics and Automation (ICRA) (pp. 3626-3633).

Bargoti, S., & Underwood, J. P. (2017). Image segmentation for fruit detection and yield estimation in apple orchards. *Journal of Field Robotics*, 34(6), 1039-1060.

Blehm, C., Vishnu, S., Khattak, A., Mitra, S., & Yee, R. W. (2005). Computer vision syndrome: A review. *Survey of Ophthalmology*, 50(3), 253-262.

Blum, A., & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In Annual Conference on Computational Learning Theory (pp. 92-100).

Borji, A., Cheng, M. M., Jiang, H., & Li, J. (2015). Salient object detection: A benchmark. *IEEE Transactions on Image Processing*, 24(12), 5706-5722.

Bradski, G., & Kaehler, A. (2008). Learning OpenCV: Computer Vision with the OpenCV library. O'Reilly Media, Inc.

Brosnan, T., & Sun, D. W. (2004). Improving quality inspection of food products by computer vision—a review. *Journal of food engineering*, 61(1), 3-16.

Bulanon, D. M., Burks, T. F., & Alchanatis, V. (2009). Image fusion of visible and thermal images for fruit detection. *Biosystems engineering*, 103(1), 12-22.

Bulanon, D. M., & Kataoka, T. (2010). Fruit detection system and an end effector for

robotic harvesting of Fuji apples. *Agricultural Engineering International: CIGR Journal*, 12(1).

Chang, K. I., Bowyer, K. W., & Flynn, P. J. (2005). Adaptive rigid multi-region selection for handling expression variation in 3D face recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)-Workshops (pp. 157-157).

Chen, R. C. (2019). Automatic license plate recognition via sliding-window DarkNet-YOLO deep learning. *Image and Vision Computing*, *87*, 47-56.

Cho, S. J., & Tropsha, A. (1995). Cross-validated R2-guided region selection for comparative molecular field analysis: A simple method to achieve consistent results. *Journal of Medicinal Chemistry*, 38(7), 1060-1066.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In IEEE CVPR'05 (Vol. 1, pp. 886-893).

Dollár, P., Appel, R., Belongie, S., & Perona, P. (2014). Fast feature pyramids for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(8), 1532-1545.

Edan, Y., Han, S., & Kondo, N. (2009). Automation in agriculture. In Springer Handbook of Automation (pp. 1095-1128). Springer.

Felzenszwalb, P. F., Girshick, R. B., & McAllester, D. (2010). Cascade object detection with deformable part models. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 2241-2248). IEEE.

Felzenszwalb, P. F., Girshick, R. B., McAllester, D., & Ramanan, D. (2009). Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9), 1627-1645.

Felzenszwalb, P. F., McAllester, D., & Ramanan, D. (2008). A discriminatively trained, multiscale, deformable part model. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 1-8).

Feng, J., Chen, J., Liu, L., Cao, X., Zhang, X., Jiao, L., & Yu, T. (2019). CNN-based multilayer spatial–spectral feature fusion and sample augmentation with local and nonlocal constraints for hyperspectral image classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *12*(4), 1299-1313.

Forsyth, D. A., & Ponce, J. (2002). Computer Vision: A Modern Approach. Prentice Hall Professional Technical Reference.

Fukushima, K. (2013). Artificial vision by multi-layered neural networks: Neocognitron and its advances. *Neural Networks*, *37*, 103-119.

Fukushima, K., & Miyake, S. (1982). Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition. In *Competition and Cooperation in Neural Nets* (pp. 267-285). Springer, Berlin, Heidelberg.

Ghiasi, G., & Fowlkes, C. C. (2014). Occlusion coherence: Localizing occluded faces with a hierarchical deformable part model. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 2385-2392).

Girshick, R. (2015). Fast R-CNN. In IEEE International Conference on Computer Vision (pp. 1440-1448).

Gomolka, Z. (2018). Backpropagation algorithm with fractional derivatives. In ITM Web of Conferences (Vol. 21, p. 00004). EDP Sciences.

Gould, S. (2012). DARWIN: A framework for machine learning and computer vision research and development. *The Journal of Machine Learning Research*, 13(1), 3533-3537.

Grandvallet, B., Zemouche, A., Boutayeb, M., & Changey, S. (2013). Real-time attitude-independent three-axis magnetometer calibration for spinning projectiles: A sliding window approach. *IEEE Transactions on Control Systems Technology*, 22(1), 255-264.

Grys, B. T., Lo, D. S., Sahin, N., Kraus, O. Z., Morris, Q., Boone, C., & Andrews, B. J. (2017). Machine learning and computer vision approaches for phenotypic profiling. *Journal of Cell Biology*, 216(1), 65-71.

Han, J., Shao, L., Xu, D., & Shotton, J. (2013). Enhanced computer vision with Microsoft kinect sensor: A review. *IEEE Transactions on Cybernetics*, 43(5), 1318-1334.

Hawkins, D. M. (2004). The problem of overfitting. *Journal of Chemical Information and Computer Sciences*, *44*(1), 1-12.

Hayashi, M., Ueda, Y., & Suzuki, H. (1988). Development of agricultural robot. In Sixth Conference on Robotics (pp. 579-580).

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 770-778).

Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, *18*(7), 1527-1554.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. Preprint arXiv:*1207.0580*.

Hinton, G., Srivastava, N., & Swersky, K. (2012). Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited On*, *14*(8).

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735-1780.

Hubel, D. H., & Wiesel, T. N. (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, *160*(1), 106.

Ji, W., Zhao, D., Cheng, F., Xu, B., Zhang, Y., & Wang, J. (2012). Automatic recognition vision system guided for apple harvesting robot. *Computers & Electrical Engineering*, 38 (5), 1186-1195.

Jones, M., & Viola, P. (2003). Fast multi-view face detection. Mitsubishi Electric Research Lab TR-20003-96, 3(14), 2.

Korine, C., & Kalko, E. K. (2005). Fruit detection and discrimination by small fruit-eating bats (Phyllostomidae): Echolocation call design and olfaction. *Behavioral Ecology and Sociobiology*, 59(1), 12-23.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems (pp. 1097-1105).

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84-90.

Ku, W., Storer, R. H., & Georgakis, C. (1995). Disturbance detection and isolation by dynamic principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 30(1), 179-196.

Kumar, P. S., Brooker, M. R., Dowd, S. E., & Camerlengo, T. (2011). Target region selection is a critical determinant of community fingerprints generated by 16S pyrosequencing. *PloS One*, 6(6), 20956.

Laguna, J. O., Olaya, A. G., & Borrajo, D. (2011). A dynamic sliding window approach for activity recognition. In International Conference on User Modeling, Adaptation, and Personalization (pp. 219-230). Springer, Berlin, Heidelberg.

Latypova, R., & Tumakov, D. (2018). Method of selecting an optimal activation function in perceptron for recognition of simple objects. In IEEE East-West Design & Test Symposium (EWDTS) (pp. 1-5).

Law, H., & Deng, J. (2018). Cornernet: Detecting objects as paired keypoints. In European Conference on Computer Vision (ECCV) (pp. 734-750).

Lawrence, S., Giles, C. L., Tsoi, A. C., & Back, A. D. (1997). Face recognition: A convolutional neural-network approach. *IEEE Transactions on Neural Networks*, 8(1), 98-113.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436-444.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., & Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Computation*, *1*(4), 541-551.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278-2324.

Li, Q., Niaz, U., & Merialdo, B. (2012). An improved algorithm on Viola-Jones object detector. In International Workshop on Content-Based Multimedia Indexing (CBMI) (pp. 1-6).

Li, Y., & Yuan, Y. (2017). Convergence analysis of two-layer neural networks with ReLU activation. In Advances in Neural Information Processing Systems (pp. 597-607).

Lienhart, R., & Maydt, J. (2002). An extended set of Haar-like features for rapid object detection. In International Conference on Image Processing (Vol. 1, pp. I-I).

Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 2117-2125).

Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In IEEE International Conference on Computer Vision (pp. 2980-2988).

Liorca, D. F., Arroyo, R., & Sotelo, M. A. (2013). Vehicle logo recognition in traffic images using HOG features and SVM. In IEEE Conference on Intelligent Transportation Systems (ITSC 2013) (pp. 2229-2234).

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. In European Conference on Computer Vision (pp. 21-37). Springer.

Liu, Z., Yan, W., Yang, B. (2018) Image denoising based on a CNN model. In IEEE ICCAR'18, pp.389-393

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), 91-110.

Lu, J., & Sang, N. (2015). Detecting citrus fruits and occlusion recovery under natural illumination conditions. *Computers and Electronics in Agriculture*, 110, 121-130.

Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In ICML, 30(1), 3.

Mavroforakis, M. E., & Theodoridis, S. (2006). A geometric approach to support vector machine (SVM) classification. *IEEE Transactions on Neural Networks*, 17(3), 671-682.

McClelland, J. L., Rumelhart, D. E. (1986). Parallel distributed processing. *Explorations in the Microstructure of Cognition*, *2*, 216-271.

McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, *5*(4), 115-133.

Minsky, M., & Papert, S. A. (2017). Perceptrons: An Introduction to Computational Geometry. MIT press.

Mizuno, K., Terachi, Y., Takagi, K., Izumi, S., Kawaguchi, H., & Yoshimoto, M. (2012). Architectural study of HOG feature extraction processor for real-time object detection. In IEEE Workshop on Signal Processing Systems (pp. 197-202).

Moeslund, T. B., & Granum, E. (2001). A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3), 231-268.

Moltó, E., Pla, F., & Juste, F. (1992). Vision systems for the location of citrus fruit in a tree canopy. *Journal of Agricultural Engineering Research*, 52 , 101-110.

Newell, A., Yang, K., & Deng, J. (2016). Stacked hourglass networks for human pose estimation. In European Conference on Computer Vision (pp. 483-499).

Nixon, M., & Aguado, A. (2019). Feature Extraction and Image Processing for Computer Vision. Academic Press.

Pan, C., Yan, W. (2019) A learning-based positive feedback in salient object detection.In IEEE IVCNZ'19.

Pan, C., Yan, W. (2020) Salient object detection based on perception saturation. *Springer Multimedia Tools and Applications*, 79 (27-28), 19925-19944.

Papageorgiou, C., & Poggio, T. (2000). A trainable system for object detection. *International Journal of Computer Vision*, 38(1), 15-33.

Patrício, D. I., & Rieder, R. (2018). Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review. *Computers and Electronics in Agriculture*, 153, 69-81.

Pehlevan, C., & Chklovskii, D. B. (2015). Optimization theory of hebbian/anti-hebbian networks for PCA and whitening. In *Annual Allerton Conference on Communication, Control, and Computing (Allerton)* (pp. 1458-1465).

Prince, S. J. (2012). Computer Vision: Models, Learning, and Inference. Cambridge University Press.

Ranjan, R., Patel, V. M., & Chellappa, R. (2015). A deep pyramid deformable part model for face detection. In IEEE International Conference on Biometrics Theory, Applications and Systems (BTAS) (pp. 1-8).

Rätsch, G., Onoda, T., & Müller, K. R. (2001). Soft margins for AdaBoost. *Machine Learning*, 42(3), 287-320.

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, real-time object detection. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 779-788).

Redmon, J., & Farhadi, A. (2017). YOLO9000: Better, faster, stronger. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 7263-7271).

Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In Advances in Neural Information Processing Systems (pp. 91-99).

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386.

Rosenfield, M. (2011). Computer vision syndrome: A review of ocular causes and potential treatments. *Ophthalmic and Physiological Optics*, 31(5), 502-515.

Sa, I., Ge, Z., Dayoub, F., Upcroft, B., Perez, T., & McCool, C. (2016). Deepfruits: A fruit detection system using deep neural networks. *Sensors*, 16(8), 1222.

Samuel, A. (1959). Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 210–22.

Sarle, W. S.(1994). Neural networks and statistical models, In Annual SAS Users Group International Conference, pp. 1538-1550.

Schalkoff, R. J. (1989). Digital Image Processing and Computer Vision (Vol. 286). New York: Wiley.

Sebe, N., Cohen, I., Garg, A., & Huang, T. S. (2005). Machine Learning in Computer Vision (Vol. 29). Springer Science & Business Media.

Si, Y., Liu, G., & Feng, J. (2015). Location of apples in trees using stereoscopic vision. *Computers and Electronics in Agriculture*, 112, 68-74.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *Preprint arXiv*:1409.1556.

Specht, D. F. (1990). Probabilistic neural networks. *Neural Networks*, 3(1), 109-118.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, *15*(1), 1929-1958.

Suykens, J. A., & Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural Processing Letters*, 9(3), 293-300.

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. (2016). Inception-v4, Inception-

ResNet and the impact of residual connections on learning. *arXiv preprint arXiv:1602.07261*.

Szegedy, C., Toshev, A., & Erhan, D. (2013). Deep neural networks for object detection. In Advances in Neural Information Processing Systems (pp. 2553-2561).

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 2818-2826).

Szeliski, R. (2010). Computer vision: algorithms and applications. Springer Science & Business Media.

Thompson, P. M., Cannon, T. D., Narr, K. L., Van Erp, T., Poutanen, V. P., Huttunen, M., & Dail, R. (2001). Genetic influences on brain structure. *Nature Neuroscience*, *4*(12), 1253-1258.

Viola, P., & Jones, M. (2001). Robust real-time object detection. *International Journal of Computer Vision*, 4(34-47), 4.

Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience*, 2018.

Wang, L., Yan, W. (2021) Tree leaf detection based on deep learning. In ISGV'21, Springer.

Wasserman, P. D. (1993). Advanced Methods in Neural Computing. John Wiley & Sons.

Werbos, P. J. (1990). Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*, *78*(10), 1550-1560.

Werbos, P. J. (2019). The New AI: Basic concepts, and urgent risks and opportunities in the internet of things. In Artificial Intelligence in the Age of Neural Networks and Brain Computing (pp. 161-190). Academic Press.

Widrow, B., & Lehr, M. A. (1990). 30 years of adaptive neural networks: Perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, 78(9), 1415-1442.

Widrow, B., Kim, Y., Park, D., & Perin, J. K. (2019). Nature's learning rule: The Hebbian-LMS algorithm. In Artificial Intelligence in the Age of Neural Networks and Brain Computing (pp. 1-30). Academic Press.

Yonaba, H., Anctil, F., & Fortin, V. (2010). Comparing sigmoid transfer functions for neural network multistep ahead streamflow forecasting. *Journal of Hydrologic Engineering*, *15*(4), 275-283.

Xiao, B., Nguyen, M, Yan, W. (2021) Apple ripeness identification using deep learning models. In ISGV 2021, Springer.

Xie, Y., Liu, L. F., Li, C. H., & Qu, Y. Y. (2009). Unifying visual saliency with HOG feature learning for traffic sign detection. In Intelligent Vehicles Symposium (pp. 24-29).

Xin, H., & Shao, B. (2005). Real-time behavior-based assessment and control of swine thermal comfort. In Livestock Environment VII, 18-20 (pp. 694).

Xuegong, Z. (2000). Introduction to Statistical Learning Theory and Support Vector Machines. Acta Automatica Sinica, 26(1), 32-42.

Yan, W. (2019) Introduction to Intelligent Surveillance. Springer

Yan, W. (2021) Computational Methods for Deep Learning. Springer

Yan, J., Lei, Z., Wen, L., & Li, S. Z. (2014). The fastest deformable part model for object detection. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 2497-2504).

Yu, F., Wang, D., Shelhamer, E., & Darrell, T. (2018). Deep layer aggregation. In IEEE Conference on Computer Vision and Pattern Recognition (pp. 2403-2412).

Zador, A., Koch, C., & Brown, T. H. (1990). Biophysical model of a Hebbian synapse. *Proceedings of the National Academy of Sciences*, *87*(17), 6718-6722.

Zhao, K. Yan, W. (2021) Fruit detection from digital images using CenterNet. In ISGV 2021, Springer.

Zhou, X., Wang, D., & Krähenbühl, P. (2019). Objects as points. Preprint arXiv:1904.07850.

Zurada, J. M. (1992). Introduction to Artificial Neural Systems (Vol. 8). St. Paul: West.