DETECT DENSE PRODUCTS ON GROCERY SHELVES WITH DEEP LEARNING TECHNIQUES

by

Li Shen

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the Degree of

Master of Science

Department of Computer and Information Technology

West Lafayette, Indiana

May 2020

THE PURDUE UNIVERSITY GRADUATE SCHOOL

STATEMENT OF COMMITTEE APPROVAL

Dr. Baijian Yang, Chair

Department of Computer and Information Technology

Dr. Wenhai Sun

Department of Computer and Information Technology

Dr. Byung-Cheol Min

Department of Computer and information Technology

Approved by:

Dr. Eric T. Matson

Head of the Graduate Program

To my family for their love and support throughout my life

ACKNOWLEDGMENTS

First of all, I want to grateful appreciation to my parents for their support and love in my life. Thank you both for giving me the strength to overcome many obstacles.

Next, I would like to sincerely thank my master supervisor. Prof. Yang, for his guidance and support throughout my master's degree, and especially for his confidence in me. I would be grateful to Prof. Sun for serving as a committee member on my thesis. Also, let me an opportunity to become his teaching assistant. I was grateful for the discussion and interpretation of some results presented in this thesis. I learned much from his insight. To Prof Ming, his comments and discussion with me were very beneficial in my completion of the thesis.

To all my friends and members from DAO2 Lab, thank you for your understanding and encouragement in my many moments that I need help. I will always remember the friendship makes my life a wonderful experience.

At last, I would like to leave the rest space for the poem that I like the most.

Do not go gentle into that good night,

Rage, rage against the dying of the light.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
RoI	Area of Interest
RPN	Region Proposal Network
SVM	Support Vector Machine
SSD	Single Shot Detector
YOLO	You Only Look Once

ABSTRACT

Object detection is a considerable area of computer vision. The aim of object detection is to increase its efficacy and accuracy that have always been targeted. The research area of object detection has many broad areas, include self-driving, manufacturing and retail stores. However, scenes of using object detection in detecting dense objects have rarely gathered in much attention. Dense and small object detection is relevant to many real-world scenarios, for example, in retail stores and surveillance systems. Human suffers the speed and accuracy to count and audit the crowded product on the shelves. We motivate to detect the dense product on the shelves. It is a research area related to industries. In this thesis, we going to fine-tune CenterNet as a detector to detect the objects on the shelves. To validate the effectiveness of CenterNet network architecture, we collected the Bottle dataset that collected images from real-world supermarket shelves in different environments. We compared performance on the Bottle Dataset with many different circumstances. The ResNet-101(colored+PT) achieved the best result of mAP and AP_{50} of CenterNet that outperform other network architectures. we proved perspective transformation can be implemented on state-of-the-art detectors, which solved the issue when detector did not achieve a good result on strongly angled images. We concluded that colored information did contribute to the performance in detecting the objects on the shelf, but it did not contribute as much as geometric information provided for learning its information. The result of the accuracy of detection on CenterNet meets the need of accuracy on industry requirements.

CHAPTER 1. INTRODUCTION

In this chapter, we provided the introduction to the previous research in object detection. We defined the research scope, research question, significance and purpose of studying issues of object detection. We also pointed out the limitation and delimitation of our thesis.

1.1 Scope

Object detection and deep learning method have been utilized in many research and real-world applications in recent years: image classification, human behaviors, and autonomous driving problems. Moreover, there are various sub-tasks related such as the face, pedestrian and skeleton detection. In early stages, object detection has achieved a 'top 5 error rates' of 2.25 percent on ImageNet challenge, which from Russakovsky et al. (2015), the result surpass the 5 percent of "top 5 error rate" in the performance. The impressive abilities of deep neural network in objection detection and feature extraction have shown the potential for future work. Therefore, researchers applied deep learning techniques to explore in many broad areas. Girshick (2015) introduced Fast-RCNN trained with a neural network that builds up the bounding box regression. It increased the detection accuracy and reduced the training time. Yolo was introduced by Redmon, Divvala, Girshick, and Farhadi (2015) which detect the target object over a dense sampling of all locations without the undergo the process of region proposal. Ioffe and Szegedy (2015) suggested batch normalization to increase learning rate and reduce training time by normalizing each training mini batch. Their methods of object detection most depend on balanced and large public datasets that aim to achieve a good performance in every scenario. But it received a drop of performance if aiming a specific application scenario.Varadarajan, Kant, and Srivastava (2019) trained a Faster-RCNN of a small grocery scenario dataset aimed at packed products. They achieved a relatively low performance with $mAP = 0.56$ at standard IoU is 0.5.

In this thesis, we presented a fine tuned CenterNet can well detect the dense objects on the market shelf. The first phase of our research is to collect and label the Bottle dataset.. The second phase is to compare the performance on detecting the dense objects between Centernet, Faster-RCNN, and YOLOv3.

1.2 Significance

Deep neural network has obtained many successes in recent years. The accuracy and speed have always improved a lot in different object detection architecture. For example, YOLO and R-CNN each has its own strength in all aspects. One of the drawbacks of them is the trade-off between speed and accuracy. YOLO is well known for its speed, but Faster-RCNN is more accurate. Therefore, we decided to use CenterNet, an anchor free detector that can both take care of speed and accuracy in our case.

In real-world situations, a human can take as long as 10 to 40 seconds to manually complete count the objects on one image with low accuracy. Thus, it is hard for human to count and audit the crowded dense products on the shelves. Therefore, the implementation of object detector in detecting the products on the shelves at retail stores and surveillance systems is necessary. The datasets of object detection are often large with a variety of instances of many object categories. To tackle the specific application scenario, we collected the Bottle dataset is indispensable. The current method from Shmelkov, Schmid, and Alahari (2017) used Faster-RCNN to detect dense objects with incremental learning that leads to a large drop in performance. Therefore, the exploration of a new object detector is significant.

1.3 Research Question

In this thesis, we try to design and integrate a solution to solve the detection issue on dense objects. We focus on using CenterNet to provide a good performance and accuracy on detecting the products on the grocery shelves and implement Faster-RCNN, YOLOv3 as the baseline detector. We choose Faster-RCNN, YOLOv3, and CenterNet as our object detector due to the statement that each of them represents the current cutting-edge detector in different stages. We can split the research question into several steps. In the first step, we use Faster-RCNN, YOLOv3, and CenterNet as our detectors to implement on Bottle dataset. In the second step, we experiment with different network architecture and fine-tune to solve the detection issue on different detectors. In this case, each object detector may face different issues on detecting the dense objects. The difference between the network architecture obtained by the different network structures and feature maps will serve as the basis for classification and localization for the detectors. In the last step, we used mean average precision metrics to evaluate each detection performance comparison on the Bottle Dataset from each detector. To validate, we depend on the annotation results demonstrate on the test result shown on the test set. The evaluation metric calculates the predicted bounding boxes intersection over union(IoU) score with the ground truth bounding boxes. To research on the contribution of color information for detecting the Bottle dataset. We convert colored images to grayscaled images and used in training and validation dataset. At last, we also calculate the accuracy of detection on our object detector.

In summary, the thesis implemented Faster-RCNN, YOLOv3, and CenterNet as object detectors on the Bottle dataset. The fine-tuned object detector would be evaluated through their performance and accuracy.

1.4 Assumptions

- CenterNet achieves the best performance on Bottle dataset compared to Faster-RCNN and YOLOv3 .
- We do not need to detect the object that has been blocked by other products or tags.
- Object detectors cannot detect the bottles on strongly skewed images.
- The accuracy of the detection on object detector can reach the requirement of industry that surpass accuracy of 93% on detecting the objects.

1.5 Limitations

- Due to time constraints, YOLOv3 was implemented parallel training on a different training and testing platform compared to Faster-RCNN and CenterNet.
- We only focus on detect the front objects on the grocery shelf.

1.6 Delimitations

- Due to the time constraints, this work will train less than 20 categories of the product on the shelf.
- Our framework does not consider real time object detection.

1.7 Definitions

Convolutional Neural Network - In deep learning, CNN is represent as convolutional neural network or ConvNet and widely used in image,video recognition, classification and natural language processing. It is a class of deep neural networks used a non-linear operation, which call convolution to extract the unique feature from the images. Moreover, CNN can using different kernels to stack the feature together. LeCun et al., mentioned that the feature of an object can be extracted after translation or rotation. In order to save the number of parameters,which lead to the property of shift invariant.(LeCun et al., 1995)

Support Vector Machine (SVM) - SVM is the abbreviation of supervised machine learning algorithm, it can be used for both classification or regression challenges. It uses a technique called the kernel trick to transform your data and then based on these transformations it finds an optimal boundary between the possible outputs.(Hearst, 1998) Basically. it does some extremely complex data transformations, then figures out how to separate your data based on the labels or outputs you've defined.

Bounding box - In object detection, a bounding box is to describe as the target location. It is a rectangular box that have x and y axis coordinates in each corner of the rectangle. When researcher annotate the data, the bounding is the rectangular box on the picture and label is the information on the bounding box.

RPN - RPN is presented as region proposal network. It is mainly used in two state object detection on Faster R-CNN that help algorithm to decide where to detect. It helps reduce the computational in overall inference process.(Zhou, 2018)

Loss Function - When doing unsupervised learning, researcher hope to minimize the difference between predictions and the ground truth. Since the output of prediction can be different compare with ground truth, to minimize the difference beweeen them is called loss function. The derivation and calculating the gradients are easy to achieve by loss function.(Zhao, Gallo, Frosio, & Kautz, 2015)

1.8 Summary

In chapter 1, we defined the research questions, pointed out the significance, scope and limitations of the thesis. In addition, the chapter provided the definitions that would be used in the research.

CHAPTER 2. REVIEW OF LITERATURE

2.1 Two-stage object detection

2.1.1 R-CNN

Two-stage object detection is constituted by two steps. The first step is applying selective search(SS) to generate a set of candidate boxes, can also describe as ”region of interest” or ROI. The second step is extracting CNN to do classification independently. Two-stage objection detection was first introduced by Girshick, Donahue, Darrell, and Malik (2013). R-CNN main workflow can be described as four steps.The figure: Regions with CNN features shows the overall workflow.

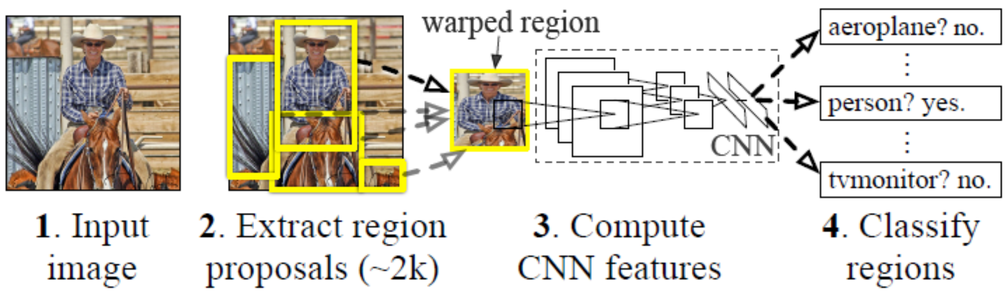


Figure 2.1. R-CNN Region Feature

- First, select input images
- From the bottom to the up of the input images, collect over 2000 region proposal
- Use CNN to compute features for each proposal
- The last step is to classify each region by using linear SVMs

The R-CNN object detection system consists of three modules that firstly implementation of R-CNN is train a classification model. To warped regions to $(N+1)$ classes, we need fine-tune the model before head. After give each image its region, train SVM classifier to determine the candidate for the objects. To improve localization performance and reduce the error, the author suggested train a bounding box regression to correct the prediction detection window. So as to learn a transformation, Using P as the input of all the transformation function, $P = p = (px, py, pw, ph)$ is the predicted bounding box coordinate and corresponding with ground truth box coordinates. Which present as $g = (gx, gy, gw, gh)$ The regression target T for the training pair is listed below:

$$t_x = (g_x - p_x)/p_w \tag{2.1}$$

$$t_y = (g_y - p_y)/p_h \tag{2.2}$$

$$t_w = \log(g_w/p_w) \tag{2.3}$$

$$t_h = \log(g_h/p_h) \tag{2.4}$$

A standard regression is shown below, it can solve issue by reduce the SSE loss with regularization:

$$\mathscr{L}_{\text{reg}} = \sum_{i \in \{x,y,w,h\}} (t_i - d_i(\mathbf{p}))^2 + \lambda \|\mathbf{w}\|^2 \tag{2.5}$$

One of the main issue with R-CNN is the training time, it takes extensive resources of time to classify 2000 region proposals, because each image creates a lot of redundant measurements. Therefore, it comes to another shortage that R-CNN cannot be implemented real time for each test image.

2.1.2 Fast-RCNN and Faster-RCNN review

To solve the limitation from R-CNN, the new implemented RCNN called Fast R-CNN was presented by Girshick (2015) Fast R-CNN made many improvements from R-CNN that purpose a special pooling layer. The original approach on R-CNN was feeding the input image to generate the region proposals to the model. To warp the region of proposal into a fixed size, RoI pooling layer can draw the input image from different vector and length. The reshaped region proposal can feed into a FCN to generate the feature map.Due to the reduced anchor proposal, the training and testing time is dramatically decreased. Table 2.1 has shown the improvement of Faster-RCNN from previous version.

Table 2.1. *Training improvement of Fast R-CNN*

	R-CNN	Fast R-CNN
Training time:	84 hours	9.5 hours
(Speedup)	1 times	8.8 times
Test time per image	47 seconds	0.32 seconds
(Speed up)	1 times	146 times

The main workflow of Fast-R-CNN has not changed much, but alter the pretrained CNN. The authors replace the pretrained max pooling layer with a RoI pooling layer. Then replace the softmax layer and last fully connected layer with a fully connected layer and a softmax layer with a (K+1) classes. The model branches generate two output layers, one is softmax estimator of k+1 classes to output a RoI discrete probability distribution. Plus a bounding box regression model predicts RoI for each of the K classes.

The main workflow of Fast-RCNN has not changed much, but replaced max pooling layers with a RoI pooling layer from pretrained CNN.The last fully connected layer with softmax was replaced by a fully connected layer with a (K+1) classes. Faster-RCNN has a similar process of its pervious version. The main contribution of this detector is the design of region proposal network(RPN) extracts predicted area to produce feature map without to use the old method on selective search. Overall, accuracy of Faster-RCNN achieved a dramatically increasing of improvement.

2.2 One-stage Objection Detection

2.2.1 YOLO:YOU only look once

YOLO was first introduced by Redmon et al. (2015). It is one stage object detector directly detects the all possible location without the region proposal network. Therefore, it is fast on testing the images compared with RCNN. There are several advantages about YOLO. First, YOLO is fast compare to Fast R-CNN. Because the author simply uses a regression problem for frame detection. Second, Fast R-CNN generates more than half of the background error compare to YOLO. The training and testing process for YOLO is used encode contextual information to deal with the input images.

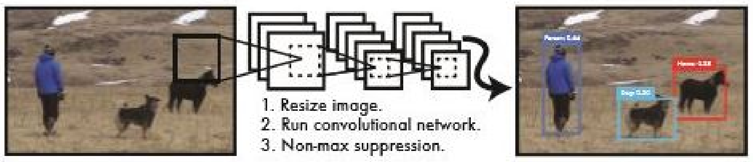


Figure 2.2. YOLO Detection System

Compare with R-CNN, the process detection system of YOLO is more simplified. Figure 2.2 shows YOLO Detection System.

- First, input images will be resized to a fixed size of 448 x 448 pixels.
- Then run a convolutional network to the input image.
- Depend on model's confidence to utilize threshold the detection result

Based on the concept of YOLO, authors introduced unified detection that divide each input image to small grids. The grid cells would be detected if it contain the feature of the detected objects. Combined grid cells from the object would form the bounding box and confidence score. The confidence scores in each grid cell present the likelihood if the cell contains an object. The author defined confidence function as 2.6.

$$Pr_(object) \times IoU_(pred, truth) \tag{2.6}$$

The notation Pr is probability of the object and IoU is the intersection over union of the detection boxes. Moreover, author also introduced $Pr(Classic|Object)$ that checks the conditional probabilities of the object in each cell and predicts one set of class probabilities. To sum up, a single image contains bounding boxes with one confidence score and coordinates to predict locations.These elements used to predict one image are functioned as $SS(5B+K)$. Moreover, the network architecture of YOLOv1 was inspired by GoogleLeNet(Szegedy et al., 2014), it formed by 2 fully connected layers combined with 24 convolution layers.The author replaced the inception module by the sequences of 1 x 1 reduction layers with 3 x 3 convolutional layers. The network of YOLOv1 is shown in Figure 2.3 below.

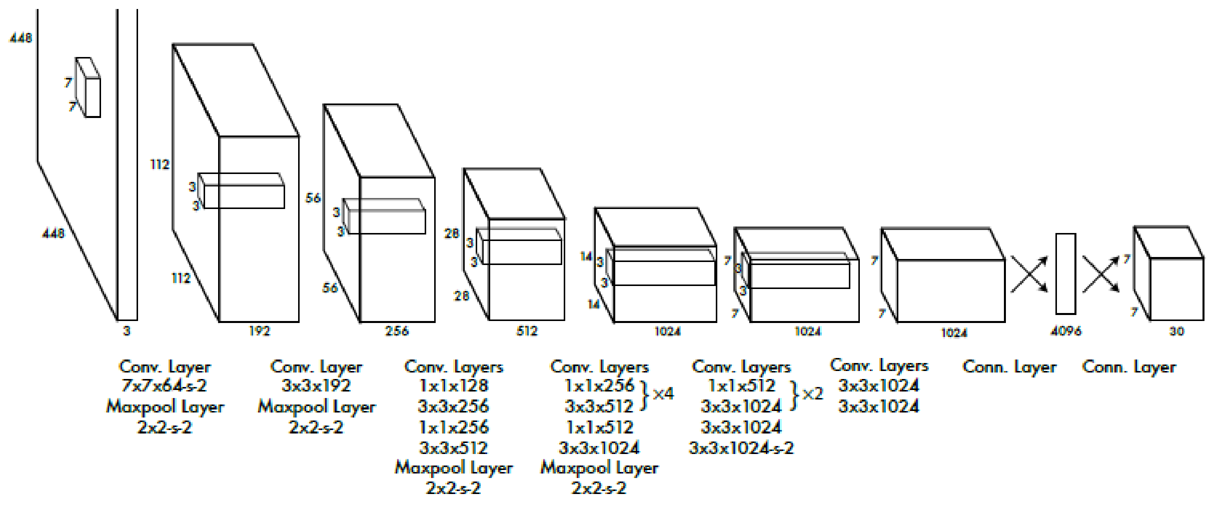

448
7
7
448
3
112
3
3
112
192
56
3
3
56
256
28
3
3
28
512
14
3
3
14
1024
7
3
3
7
1024
7
7
1024
4096
7
7
30
Conv. Layer 7x7x64-s-2 Maxpool Layer 2x2-s-2
Conv. Layer 3x3x192 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x128 3x3x256 1x1x256 3x3x512 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x256 3x3x512 }×4 1x1x512 3x3x1024 Maxpool Layer 2x2-s-2
Conv. Layers 1x1x512 3x3x1024 }×2 3x3x1024 3x3x1024-s-2
Conv. Layers 3x3x1024 3x3x1024
Conn. Layer
Conn. Layer

Figure 2.3. YOLO Network Architecture

The author described the training as following: At first, they used the ImageNet 10000 class competition dataset to pretrain the first 20 convolution layers, the input size is 224×224. Due to the the input size of the image need to increase from 224 to 448 to detect more detailed visual information. The loss function for YOLOv1 calculated by classification loss combined with localization loss. The classification loss calculated the score at each cell was present as the probabilities of each object that possible to be detected. The localization loss calculated the errors from the bounding box overlap with predicted anchors. To intergrate, the calculation of the confidence score added classification loss and localization loss to form the loss function of YOLOv1. The function is listed below.

$$\mathscr{L}_{\text{loc}} = \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 + (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \tag{2.7}$$

$$\mathscr{L}_{\text{cls}} = \sum_{i=0}^{S^2} \sum_{j=0}^{B} \big(\mathbb{1}_{ij}^{\text{obj}} + \lambda_{\text{noobj}}(1 - \mathbb{1}_{ij}^{\text{obj}})\big)(C_{ij} - \hat{C}_{ij})^2 + \sum_{i=0}^{S^2} \sum_{c \in \mathscr{C}} \mathbb{1}_{i}^{\text{obj}} (p_i(c) - \hat{p}_i(c))^2 \tag{2.8}$$

$$L = L_{\text{loc}} + L_{\text{cls}} \tag{2.9}$$

Compare the result of YOLO v1 with other objection detection algorithm(Using PASCAL VOC 2007).YOLO has many advantages compare with others. At first, YOLO has 4.75% background errors and Fast R-CNN has 13.6% background error. The result showed YOLO has attained a much lower number of background errors at a much faster detection speed. As YOLO can deal with 45 frames per second, which it is better than real-time. The limitation of YOLOv1 reveals that each grid cell can only predict one class object, which can cause undetected object. YOLO is hard to lo localize small objects or groups of small objects. This is the main reason for YOLO has relatively high errors in localization.

2.2.2 YOLOv2

In order to solve the issue with YOLOv1 made lower recall values and high value of localization errors compared to other region proposal based methods. YOLO9000 was introduced by Redmon and Farhadi (2016). Which is build on top of YOLOv2 and trained with top 9000 classes with COCO detection dataset from ImageNet. YOLOv2 implements many modifications based on the shortage of YOLOv1. First, Batch Normalization has been implemented on all the convolutional layers, they result more than 2% improvement in mAP. Which leads to significant improvement over convergence. Another improvement is add a High Resolution Classifier. It fine tune the base model with a higher resolution image from 224x224 to 448x448 that improves the detection performance. This is also a concern for me when I read the YOLOv1, because pertained input images have different resolution to training input images.

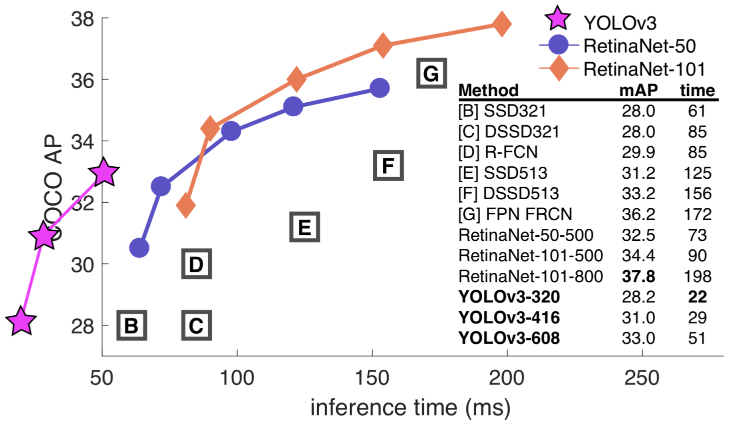
From my perspective, one of the most important improvements for YOLOv2 did not predict the position of the bounding boxes from the images of a fully-connected layers that cause a relatively low recall. YOLOv2 removed the fully connected layer by replacing the anchor boxes. Add anchor boxes success leads to a dramatic increase in recall from 81%to 88%. Authors also use clustering dimension different with other methods. They use K-mean clustering on the training data for anchor box dimension. Instead of using the same method of Fast-R-CNN, because the hand picked can not automatically find good priors. Thus, the anchor produced from clustering achieve a better averaged IoU score with fixed number of box.The distance matrix is showed below.

$$\mathrm{D}(\mathrm{box}, \mathrm{centroid}) = 1 - \mathrm{IOU}(\mathrm{box}, \mathrm{centroid}) \tag{2.10}$$

Another issue from YOLOv2 is model instability, when implemented the anchor boxes on YOLOv2. The reason is the model of bounding box prediction that formulated by YOLOv2 will not miss the center location of the predicition. If the prediction can be set at any part of the image. The model training can cause unstable. Overall, YOLOv2 has a improved a lot in many aspect and become a better model that compared with pervious version. The model was trained with Pascal Voc with dataset obtained 78.6% mAP with 40 frame per second. Moreover, the model achieved decent detection results with 2-10 times faster. The model results on COCO dataset have mAP scores of 44% for an IoU is 0.5, 19.2% for an IoU is 0.75 and 21.6% for its primary challenge metric(0.5:0.95) mAP score.To sum up, the authors introduced YOLOv2 by implementing many modifications to make its prediction more accurate and faster.

2.2.3 YOLOv3

Later, YOLOv3 was introduced by cite DBLP: yolov32018 that with an incremental improvement of the last version. The authors made many changes to make their model become better. The first improvement was bounding box prediction.YOLOv3 predicted the object score by using logistic regression of each bounding box, and the score should be one. From my understanding, score of one means bounding box is completely overlapped over the ground truth object. It will only predict 1 bounding box over 1 ground truth object. Next improvement is the class prediction which the approach is done to approach multi-label classification. The authors used an example to address this solution, we were able to label the object with both named label. YOLOv3 had 3 different scales to predict the bounding boxes.The authors used the first 2 layers of feature map with unpsampled in 2 times to support prediction across 3 scales. The method helped YOLOv3 achieve more useful feature information from fine grained information and upsampled features.As the result from YOLOv3, showed in figure 2.4 below.



Method	mAP	time
[B] SSD321	28.0	61
[C] DSSD321	28.0	85
[D] R-FCN	29.9	85
[E] SSD513	31.2	125
[F] DSSD513	33.2	156
[G] FPN FRCN	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	**37.8**	198
YOLOv3-320	28.2	**22**
YOLOv3-416	31.0	29
YOLOv3-608	33.0	51

Figure 2.4. YOLOv3 result

To sum up, YOLOv3 outperform SSD, and worse than RetinaNet. But the way more fast. In my opinion, this model is suitable for most cases if speed is important or needs to be real-time. It has decent accurate as well.

2.3 Anchor free methods

The most successful object detection algorithms like YOLO, R-CNN all rely on anchors to refine their final detection location. Although one stage object detection algorithm is faster than two stage object detection. However, YOLOv3 still has 9 different sizes of anchor boxes and need NMS to reduce them, the speed and performance of one stage object detection are still limited. Less anchor leads better speed but lower the accuracy, and vice versa. Therefore, many new work is trying a different method: design anchor free object detection algorithms. Both of CenterNet and CornerNet are based on Keypoint detection.

2.3.1 CornerNet

CornerNet is one of the representative works of keypoint detection. Law and Deng (2018a) have published paper CornerNet: Detecting Objects as Paired Keypoints to introduce their new work. They think there is one major drawback of using anchor boxes. First, a large set of anchor boxes was required for the anchor-based model training. Because the anchor boxes need to overlap with ground truth boxes. While only a few anchor boxes could achieve the job, it slows down the training process. Due to the imbalance between positive and negative anchors. (Law & Deng, 2018b) To solve this issue, Law and Deng (2018a) detect an instance by using the top-left corner and the bottom-right corner of the bounding box: a pair of key points. The figure shows the workflow of CornerNet. At last, there are specific corner pooling modules for every predicts modules and predictions of the heatmaps, embedding, and offsets.

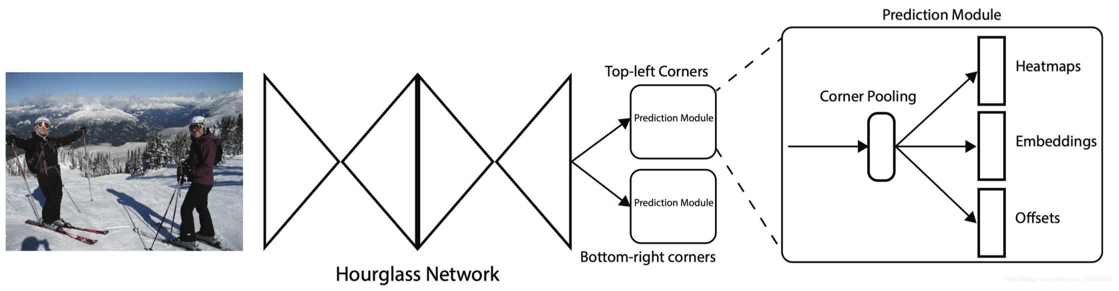


Figure 2.5. Hourglass Network

Given image go through a process to Hourglass Network is the first step of the workflow. It is the backbone network for CornerNet. The hourglass network was the solution for human pose estimation tasks in the beginning. As the paper mentioned, the authors used stride 2 to decrease feature resolution, instead of using max-pooling. The feature resolution got 5 time reductions for feature resolutions and an extension on feature channel from 256, 384 to 512 (Newell, Yang, & Deng, 2016). The next step of the workflow is the prediction module. The authors utilized corner pooling to figure out whether the pixel is the top left or bottom right corner. Which took the maximum value for each channel in two different directions. Max values are represented as red dots and directions represented as red lines. Then added two maximum values together as the blue dot shows in Figure.

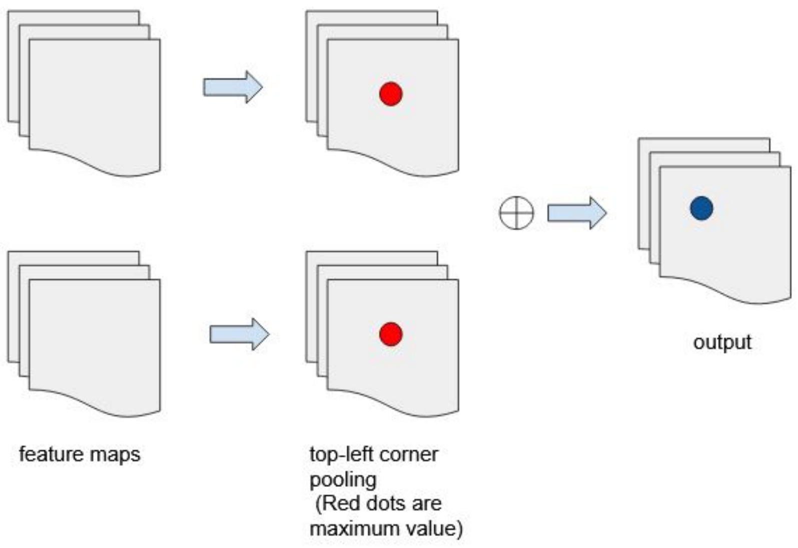


Figure 2.6. Corner pooling

The result evaluated CornetNet with the MS COCO dataset Since corner pooling is the key framework for CornerNet. In order to understand its contribution to overall performance for CornerNet. The authors discarded the corner pooling and remained the same parameter numbers still to perform another network. The result showed significant improvement when using corner pooling as the fundamental component.

	AP	AP^{50}	AP^{75}	AP^{s}	AP^{m}	AP^{l}
Corner pooling	36.5	52.0	38.9	17.6	38.7	48.8
Corner pooling	38.5	54.1	41.1	17.7	41.1	52.5
Improvement	+2.0	+2.1	+2.2	+0.1	+2.4	+3.7

Compare with other detectors, one-stage detectors, and two-stage detectors, CornerNet has better results than the former ones and more competitive results than later ones.

2.3.2 CenterNet

Another method uses combine the idea of center-based and corner-based methods. Duan et al. (2019) introduced CenterNet detector that detects an object with a single point. Keypoint represent as center point of the object. CenterNet combined the idea with corner-based methods that predict the bounding boxes by pairs of keypoints from each corner at first, then predict its center probabilities. In my opinion, the CenterNet from Duan's paper is based on the change of the CornerNet. Which they have similar workflow.

Another object detector proposed by the paper ”Objects as points”. It is written by Zhou, Wang, and Krähenbühl (2019) that detect the object by using the estimation of keypoint. The size of the bounding box is determined from the keypoint feature at the center. There are around three main reason that CenterNet has been created. Firstly, CenterNet can estimate multiple object properties and has a good speed-accuracy trade-off. Secondly, it does not have NMS with post-processing after training. Based on author's result, they added NMS as a post-processing that received a minor impact. The main contribution of the paper provides a much simpler and efficient object detector that only need find single point at their bounding box center. To train a model, user need feed the input dataset to a fully convolutional network(FCN). FCN can generate a heatmap that is the center of the object. Model used standard dense supervised learning that fast and without NMS for post-processing.

The result shows their CenterNet with Hounglass network as their backbone network has a good trade-off for COCO dataset. The method outperform other object detection algorithms.

2.4 Perspective Transformation

To convert angle images to non-angled images, perspective transformation need found four points from the original image to produce a new images. The viewpoints of the image would be changed into another plain once perspective transformation has been implemented. Wang et al. (2019) introduced usage of perspective transformation to achieve data augmentation. Their technique called perspective transformation data augmentation (PTDA) can automatically generate new data with their relevant annotations. The experiment is based on several datasets and used Faster-RCNN as their detector.

The result according to the author's result, PTDA has advantages to improve the detection performance and also add robustness of the detection model.

2.5 Current Objection Detection on Market shelves

Object detection on market shelves was first investigated from Merler, Galleguillos, and Belongie (2007).The authors propose a system based on local invariant features. Their result performance of experiments is already outdated in terms of speed and accuracy. But They created GroZi-120 database, which provides many image of grocery product to use.

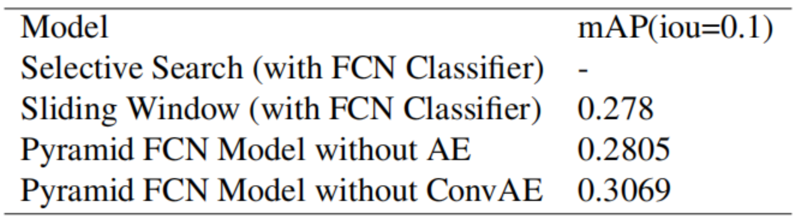
Follow highlight issues from Merler et al. (2007). Tonioni, Serra, and di Stefano (2018)proposed a deep learning pipeline in object detection on store shelves. It provide a fast and accurate approach to recognize the product on the shelves. They divide it by three parts.

- The first step, the authors used ScaleNet(CNN based method) to agnostic each class of products that identify them appear on the shelf image.
- Secondly, the authors use K-NN similarity search to embedder the reference images.
- Thirdly, the authors combined different methods to implement a final refine step. Which remove false detections and increase the accuracy in similar products.

Small objects and different camera view of our data cause localization issue.Varadarajan and Srivastava (2018) introduced a weakly supervised learning method to localize the product on grocery shelves. The authors used a simple Fully Convolutional Network(FCN) with RefineNet (G. Lin, Milan, Shen, & Reid, 2016) module to predict object localization. The dataset of this paper considers 10 brands of cigarettes on the market shelves.

In FCN architecture, authors take pretrained layers of VGG11 but remove all FC layer instead. Also add a single layer to get N times 1 times 1 output. The RefineNet module used to refine the output generate from FCN, then localize the discriminative features for each object. The methods from Varadarajan and Srivastava (2018) gives us a better idea on how we construct our network. We going to use autoencoder to extract our categorical feature map has similarities compare to their method. But their method still has weaknesses. As they used to sliding windows as their baseline. Which they use pyramid method compared their result with the traditional method. The result is shown in table 2.2 that did not have a significant improvement.

Table 2.2. *Result of FCN+ConvAE architecture*



Model	mAP(iou=0.1)
Selective Search (with FCN Classifier)	-
Sliding Window (with FCN Classifier)	0.278
Pyramid FCN Model without AE	0.2805
Pyramid FCN Model without ConvAE	0.3069

Another way to help solve localization issue on a small object is to determine shelf boundaries. Varol and Kuzu (2015) introduced a shelf boundary line to count the number of products and number of shelves. Which based on the shelf boundary, segmentation and localization can be better detected as shelf boundary line displayed on the image.

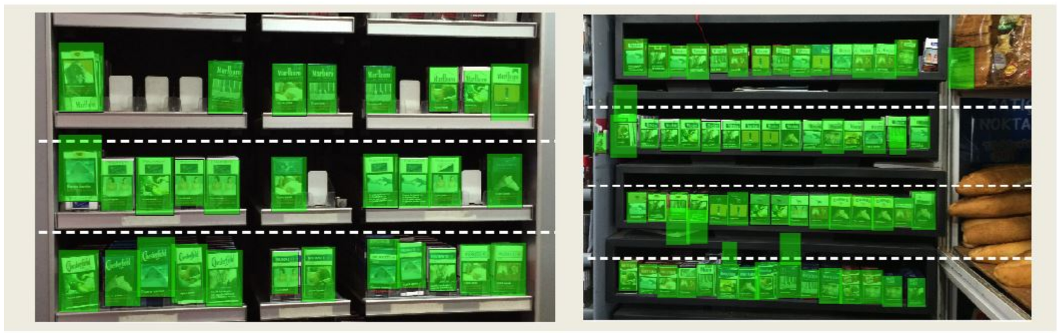


Figure 2.7. Market Shelf detection

The paper from Eggert, Zecha, Brehm, and Lienhart (2017) applied Faster R-CNN in improving small object detection to detect the company logo. Since we aim to detect the products on the grocery shelves, the approaches inside the article are very similar to our approach. The authors addressed that company logos are usually small to be detected, which requires more precise localization than normal objects to be correctly detected. Also, the confidence scores of small objects are relatively lower than the scores of large objects. To solve the issue, the authors used a different scale of the RPN on different feature maps. After the experiment result, they concluded the detected small objects on earlier feature maps are more accurate. Moreover, the author observed that the filter in the dense layer is suitable for various scales. The result of their work has been shown in figure 2.8.

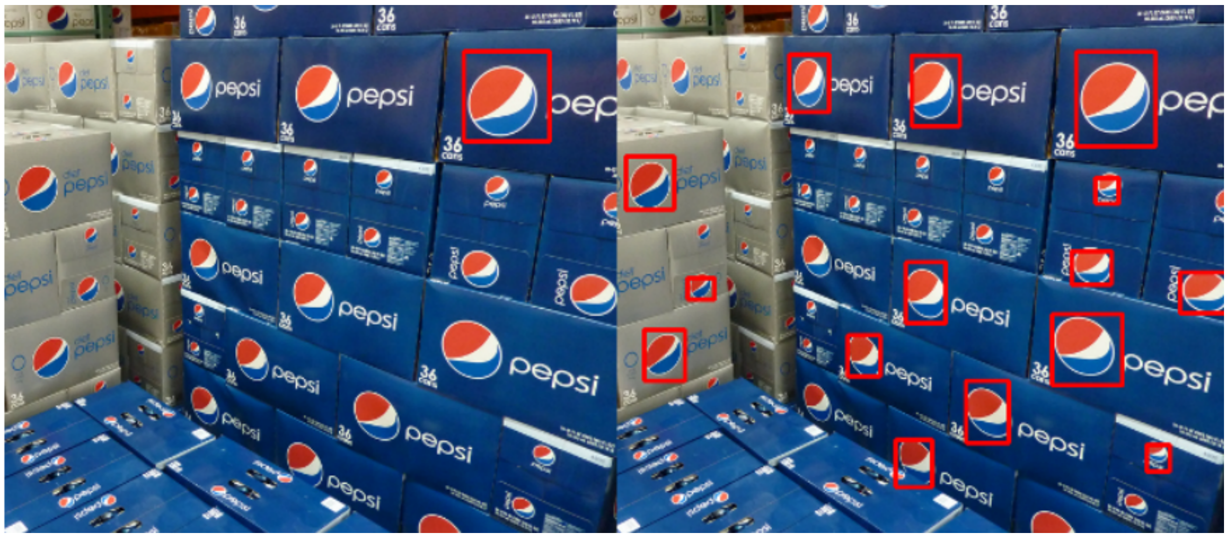


Figure 2.8. Small object detection to detect the company logo

One of the ideas we thought before was to detect the brand of each product on the market shelves. Instead input the full image of a bottle, the brand can help us successfully detect each different kind of products from different brands. But there are also limitations to this method. First, the objects from each bottle may contain multiple brands. Which means that an object may be recognized as two or more. In addition, the products on the shelf may be placed at different angles.There could be no brand logo on the back of the product.

2.6 Summary

Chapter 2 summarized the previous work of object detection algorithms including one stage, two stage and anchor free object detection algorithms. By reviewing those work, we can learn their idea to utilize on our idea of the thesis. It also states some improvements from recent try out and how to combine different framework to generate a new network architecture. At last, we can use some previous work to compare the results of the experiment.

CHAPTER 3. RESEARCH METHODOLOGY

In this chapter, we introduced three object detectors that used previous research and demonstrate how to implement them on this thesis. To detect the small object on the shelves, we will describe each object detection architecture and loss function.

3.1 Real-world datasets

We collected real-world dataset. The dataset concentrated on the purpose of object detection. We collected the data from different supermarkets in local West Lafayette. In the first step, we composed different kinds of products and labeled them. The images of the data would be taken from around 2 meters away from the market shelf. The image may contains more than 50 small objects with different angled. In total, the Bottle dataset contains 20 classes. They are listed in Table 3.1. We labeled all the images during data preparation.

Table 3.1. *Label list*

Coke_2L	Coke_small	Coke_Dietsmall	Coke_Diet	Coke_Diet2L
Coke	Coke_Zero2L	Coke_Zerosmall	Coke_Cherry2L	Sprite_small
Sprite_2L	Coke_Zero	Pepsi_2L	Pepsi	Pepsi_Diet
Pepsi_Diet2L	Coke_SmallDecaf	Coke_Decaf2L	Abnormal	Sprite

3.1.1 Randomness of placement

The products in the supermarket environment are usually placed depending on certain rules. To make the data as close as the real-world situation. The images of the Bottle dataset maintained their original place of order and not to place the products on our own. The objects on the shelf may contain the non-labeled objects which would not be learned during the training phase but would be executed during the test phase. See Figure 3.1.

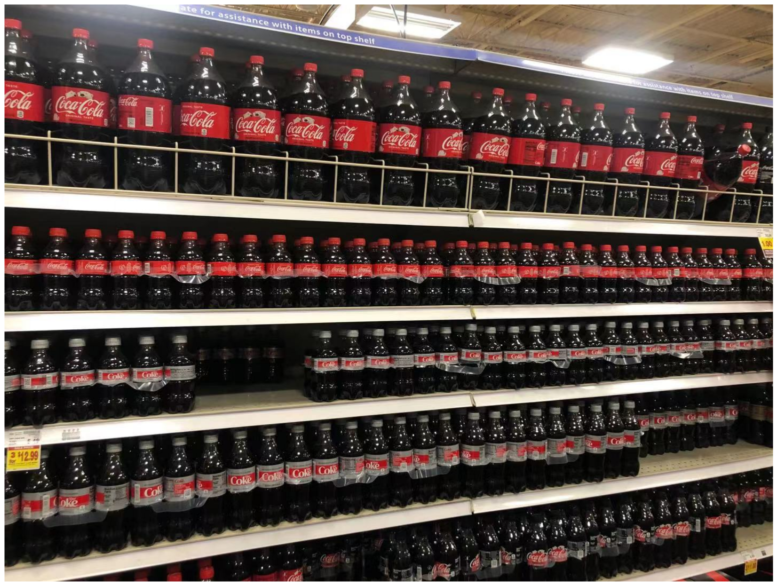


Figure 3.1. Shelve Example

3.2 Data Preparing

For better serving our goal to detect the small objects on the shelves. The dataset is collected by taking product shelves pictures containing bottles with varying backgrounds in local West Lafayette supermarkets and grocery stores. The bottle on the image is displayed as different sizes, colors, angles, different surfaces, and illumination. Bottle Dataset contains 282 of labeled colored images. We also convert all labeled colored images to grey images to discuss object detection with the importance of color and shape on bottles. Which in total to train CenterNet, we have 584 shelves of images combined with Colored images and grey images. The original labeled format is XML file, can be used as an annotation on Faster-RCNN. To use annotation on YOLO and CenterNet, we need to convert XML file to other formats. The ground truth of the Bottle dataset is manually labeled, then we used a python script to randomly selected 20% of bottle dataset as validation data, another 30% of bottle dataset as testing dataset. The rest 50% of bottle dataset as the training dataset.

We used a graphical annotation tool called Labelimg to draw bounding box and label classes of each object that each image saved an Extensible Markup Language(XML) file contains the coordinates information of the annotation. XML file is used as the annotation of Faster-RCNN that can be directly implemented for training, validation and test. To use annotation on Yolo, we have to transfer.XML files to .txt files. The format of .txt files include object coordinates and object number on the image (object-class, x, y, width, height). To use annotation on CenterNet, we need to convert .XML files to .json files. Note each of three object detectors used a unique format of annotation, and the format of the annotation does not influence the quality of the annotation.

3.3 Object Detector: Faster-Rcnn

Faster-RCNN is composed by three independent neural networks. They are Feature Network, Region Proposal Network (RPN), and Detection Network that each play important role. The first step of RPN is the input image goes through a convolution neural network (CNN), the output is the set of convolutional feature map on the last layer. RPN Combined with one-layer network of region proposal and classifier to identify the bottle class. For training phase in the RPN, it will generate 9 different size anchor boxes that remapped back to the image. Therefore, more than thousands of anchor boxes will be generated in the image. Then after a feature map processed through sliding window, a 256-dimensional vector is re-sampled before feed into two FCN layer. A box regression layer (rag) computes the box offset and a box classification layer (cls) computes the confidence score that makes the total RPN output per position to 4k coordinates plus 2k scores. The region proposal network showed as Figure 3.2.

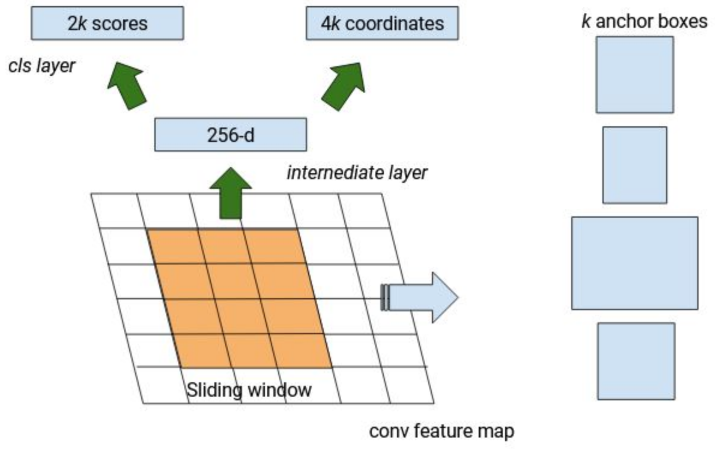


Figure 3.2. Region Proposal Network

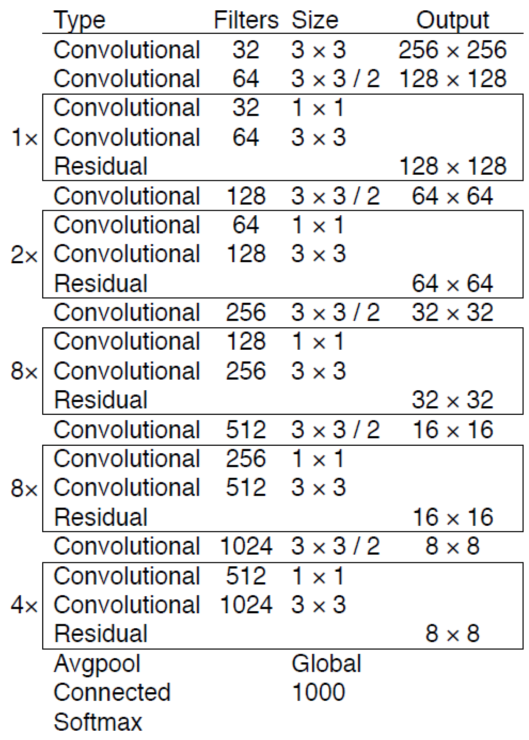
In our case with Bottle dataset, If the anchor boxes with the highest intersection overUnion(IoU) over a ground-truth bounding box or anchor has IoU higher than 0.8 with a ground-truth bounding box. We assign a positive label to it. Anchor with IoU lower than 0.2 overlaps with ground-truth bounding boxes that we will assign a negative label. The equation is the loss function.

$$\begin{aligned} L(\{p_i\},\{t_i\}) = \frac{1}{N}\sum_i L(p_i,p_i^*) \\ +\lambda\frac{1}{N}\sum_i p_i^* L(t_i,t_i^*). \end{aligned} \tag{3.1}$$

After region proposal, we use Non-Maximal Suppression (NMS) to reduce redundancy. Since regions can be highly overlapped with each other and we have many small objects that are similar to each other on one image. To measure the performance, we predict a bounding box compare with the ground truth boxes to calculate the Mean Average Precision (mAP). We discussed it in Section 3.9 Evaluation Criterion.

3.4 Object Detector: YOLOv3

To train Bottle Dataset use YOLO, we used a PyTorch implementation of YOLOv3 to support training and evaluation. In this framework, we utilized a fully convolutional network (FCN). But instead of using ResNet101 as our backbone network to feature extractor. We utilized DarkNet-53 that total has 53 convolutional layers that contain batch normalization with Leaky ReLU activation. The architecture of DarkNet-53 is shown in Figure 3.3 .



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 3.3. DarkNet-53 Network Structure (Redmon & Farhadi, 2018)

The DarkNet-53 did not use any of the pooling method to use as downsampling the feature maps. It used a convolutional layer with 2 strides. For bounding box prediction, YOLOv3 used the same system as YOLOv2 did. Dimension clusters of YOLOv3 is used to predict the bounding box. The anchor boxes will be generated that predict 4 coordinates for each side of the box. It usually presents as t_x, t_y, t_w, t_h. The offset of the cell is (c_x, c_y) from the top left of the picture. Class prediction on each bottle, we used independent logistic classifier and binary cross-entropy loss to deal with overlapping labels. In our case, we only going to detect the most front bottle that has been placed on the shelves. YOLOv3 also uses NMS to predict the best bounding box and remove any bounding box that probability is lower than its given NMS threshold. Then, NMS selects the bounding boxes with the highest detection score from the IOU value high than IOU threshold.

$$\begin{aligned} b_x &= \sigma(t_x) + c_x \\ b_y &= \sigma(t_y) + c_y \\ b_w &= p_w e^{t_w} \\ b_h &= p_h e^{t_h} \end{aligned} \tag{3.2}$$

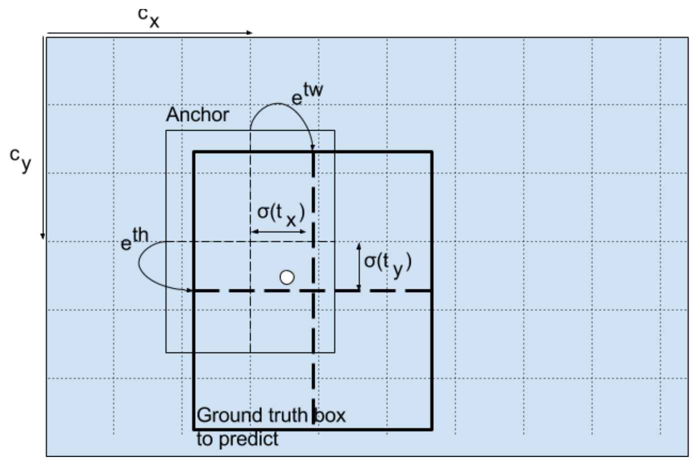


Figure 3.4. Bounding boxes with Location Prediction

3.5 Object Detector: CenterNet

To train Bottle Dataset by using CenterNet, we implemented CenterNet from Zhou et al. (2019). The overall workflow is quite simple. Firstly, we feed Bottle Dataset as the input image to a fully convolution network that can generate a heatmap for us. The peak of the heatmap is related to the center of the object. Then image feature at each center point predicts the height and weight of the bounding box. In this framework, we used ResNet18, 101 and Hourglass as fully-convolutional encoder-decoder networks to generate the heatmap.

$$I \in R^{W \times H \times 3} \tag{3.3}$$

$$\hat{Y} \in [0,1]^{\frac{W}{R} \times \frac{H}{R} \times C} \tag{3.4}$$

Equation 3.3 is an input image and W represents width, H represents height. The aim is to produce a heatmap with the keypoint on the object represents as equation 3.4. The idea to train the keypoint prediction network is from Law and Deng (2018a). The ground truth keypoint is represented as $p \in \mathscr{R}^2$ of class c that computes a low-resolution equivalent $\tilde{p} = \lfloor \frac{p}{R} \rfloor$. Function 3.5 is a logistic regression combined with focal loss from T. Lin, Goyal, Girshick, He, and Dollár (2017).

$$L_k = \frac{-1}{N} \sum_{xyc} \begin{cases} (1-\hat{Y}_{xyc})^{\alpha} \log(\hat{Y}_{xyc}) & \text{if } Y_{xyc} = 1 \\ (1-Y_{xyc})^{\beta} (\hat{Y}_{xyc})^{\alpha} & \text{otherwise} \\ \log(1-\hat{Y}_{xyc}) & \end{cases} \tag{3.5}$$

$x_1^{(k)}, y_1^{(k)}, x_2^{(k)}, y_2^{(k)}$ is the four points of the bounding box, the K represent as object and category c_k is the category. Function 3.6 shows how to calculate the center point of the object.

$$p_k = (\frac{x_1^{(k)} + x_2^{(k)}}{2}, \frac{y_1^{(k)} + y_2^{(k)}}{2}) \tag{3.6}$$

The offset loss function is shown at equation 3.7, which all classes use one offset with an L1 loss.

$$L_{off} = \frac{1}{N} \sum_{p} \left| \hat{O}_{\tilde{p}} - \left(\frac{p}{R} - \tilde{p} \right) \right| . \tag{3.7}$$

Figure 3.5 is the model diagrams of CenterNet, (a) is Hourglass Network that author used it in CornerNet Law and Deng (2018a) as well. (b) is ResNet network with transpose convolution that upsampled the feature map. (c) is their DLA-34 and (d) is the modified DLA-34 that from the bottom layers to adds skip connections.

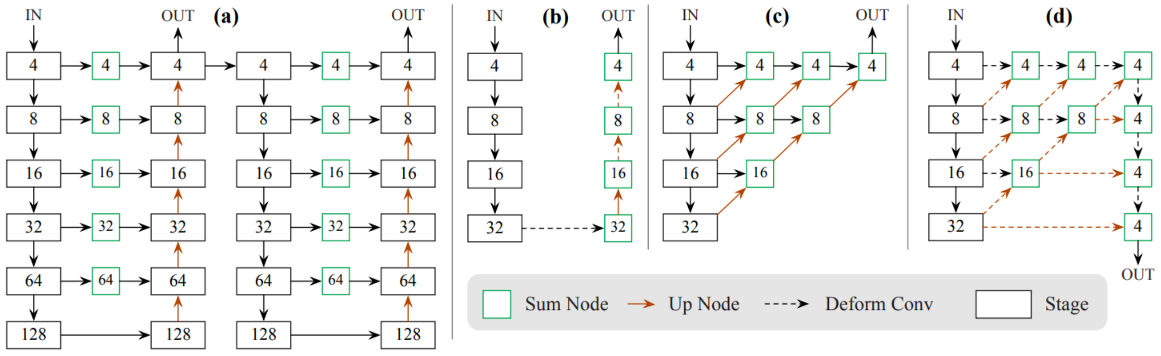


Figure 3.5. CenterNet Network Structure

To evaluate CenterNet performance on our Bottle Dataset. We use average precision over all IOU threshold(AP) and AP at IOU thresholds with 0.5 and 0.75. Further information will be explain in the evaluation section.

3.6 Categorical Anchor Generator

The encoder and decoder will adapt to an Autoencoder as figure shows below.

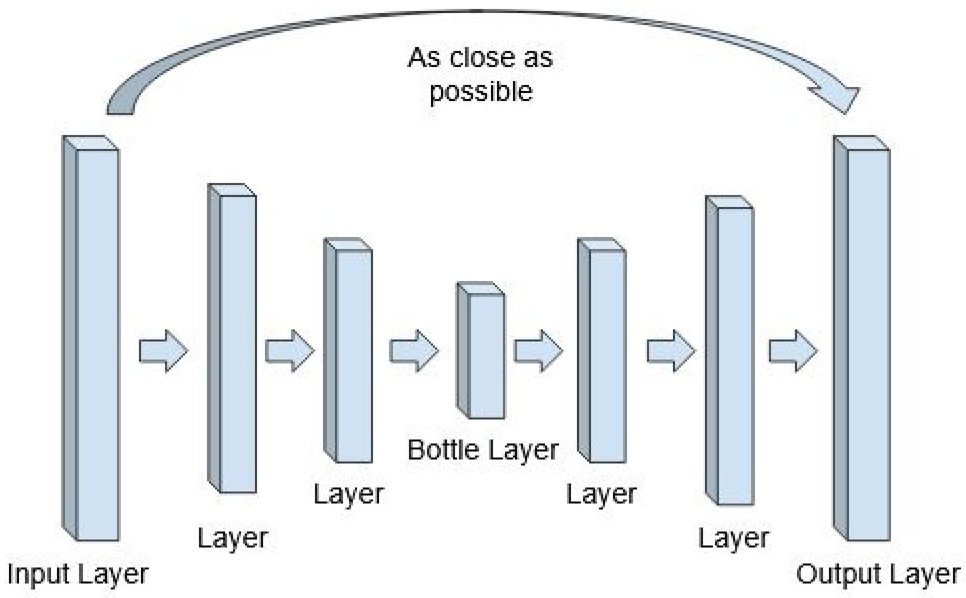


Figure 3.6. Categorical Anchor Generator

Categorical Anchor Generator(CAG) replaced fully connected(FCN) layers of convolution layers, which we can achieve by extending the basic structure of simple auto-encoder. Which we borrow the idea from convolution autoencoder. There are two processes of CAG is encode and decode. The input images will be processed through an encoder, the purpose of this that compress your input data with the encoder. The left two layers downsampling and extracting the global feature of the image. The middle layer worked as bottleneck. The right two layers used as upsampling and transpose convolution layers. The structure of CAG is quite simple, it has been constructed by input layer, output layer, and hidden layer. The output layer is used to reconstruct the input that means the loss of the output and input should be as small as possible. In our network, we will extract the embedded features from the bottle layers as our categorical anchor. Our CAG is structured in encoder layers and decoder layers. The encoder layer is presented respectively as $fW(*)$ and decoder as $gU(*)$. The aim of each layer is to find extract feature maps of each input image. We minimize the mean squared errors (MSE) from all samples generated from inputs and outputs. We defined function in 3.8.

$$min_{\mathbf{W},\mathbf{U}} \frac{1}{n} \sum_{i=1}^{n} \|gu(fw(x_i)) - xi|_F^2 \tag{3.8}$$

In our generator, x and h are vector are number of vectors. σ is the activation function. After training process, h is the code generate from the embedding layer. It is a new representation of input sample. Then h can be fed into another object detection architecture. The function of convolutional autoencoder is define as

$$fw(x) = \sigma(x * w) \equiv h \tag{3.9}$$

$$g\mu(h) = \sigma(h * U) \tag{3.10}$$

3.6.1 Network architecture

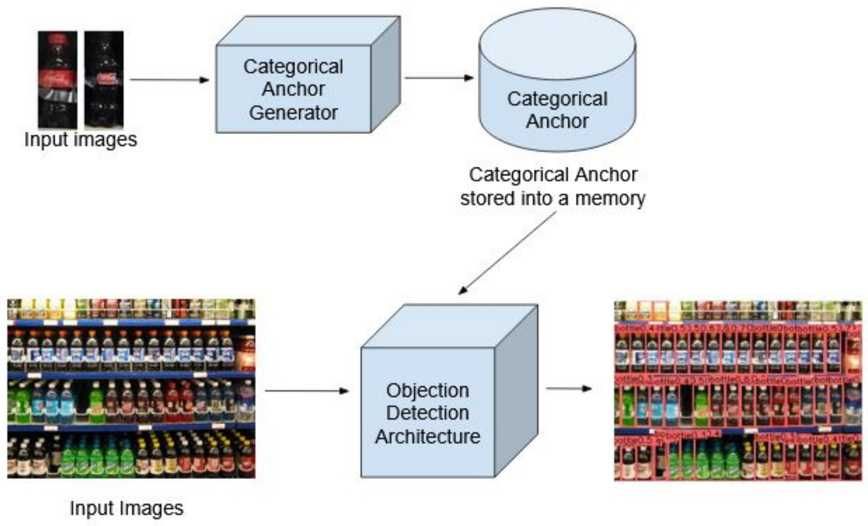

Input images
Categorical Anchor Generator
Categorical Anchor
Categorical Anchor stored into a memory
Input Images
Objection Detection Architecture

Figure 3.7. Network Architecture Workflow

To start, we design a categorical anchor generator to extract the information on categorical anchor. The input images are the products we going to detect that we define different product as an individual class. (For example, coke and diet coke will be defined as two different classes.)All input image remains non-labeled at this stage, the categorical anchor will be stored into a memory that contains the feature information about the product. The memory will be implemented as a pre-knowledge while we start training our object detection architecture. Moreover, it can be applied to assist the prediction of the detection process.

3.6.2 Unpooling and Deconvolution

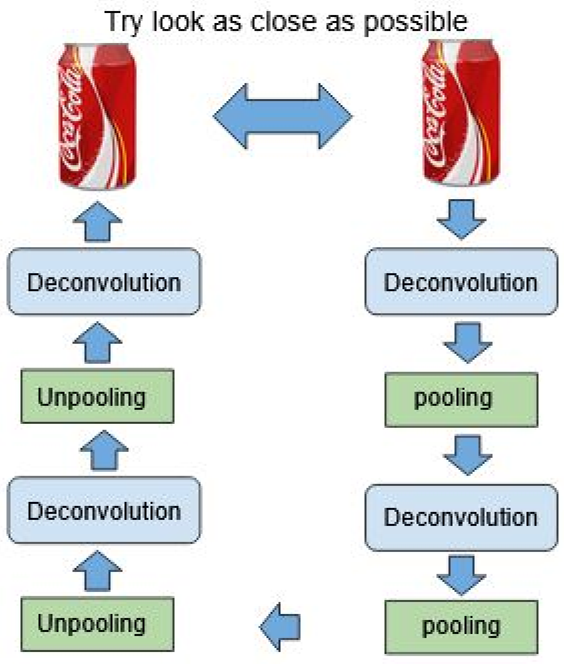


Figure 3.8. Unpooling and Deconvolution

The process of encode and decode can be understand as an inverse function. The right part of the graph shows the process of encoder continuously extracts features by using convolution and pooling. Which doing the process of dimension reduction. The left part of the graph shows the process of decoder need unpooling and deconvolution to increase the reducted dimension.

The pooling process is actually a dimensionality reduction process. We assumed the size of input image is formed by 32×32 pixels and pooling size is 2×2 pixels. After pooling process, the size of the image is $16x16$ pixels. Which equivalent to replacing the adjacent pixels with max-pooling. Unpooling process need to change the 16×16 pixels image to 32×32 pixels image. Also during the pooling process, the maximum value of the 2×2 pixels has been marked that will be restored during the unpooling process.

3.7 Evaluation Methods

To validate the performance of each object detector on Bottle Dataset. We depend on the results displayed by annotations generate on the demo detection. When the objects have been correctly recognized that intersection with the ground-truth boxes, we score a correct recognition to label the objects. In our case, We set the correct recognition is when the object has been correctly labeled with bounding box and its intersection over union(IoU) overlap with ground truth boxes are high than 0.5 average precision(AP). Mean average precision (mAP) and average precision(AP) both are a popular metrics in evaluating the performance of deep object detectors.AP calculates the value of average precision with the value of recall over 0 to 1, and mAP is calculated from all categories with their average AP value. To interpret, it means mAP is averaged over all class of objects. In our case, we will use mAP. AP to test the performance of Faster-RCNN, YOLOv3 and CenterNet, include AP at IoU thresholds 0.5 and0.75. We will test accuracy of detection on CenterNet based on the demo result.

3.7.1 Intersection over union(IoU)

Before we calculate our AP, we need to understand Intersection over union(IoU). IoU calculate the overlap between two bounding boxes area. It is used to measure the result of the bounding box overlap with the ground truth box. In our case, we set IoU threshold over 0.5 as a true positive, and below 0.5 is false positive.

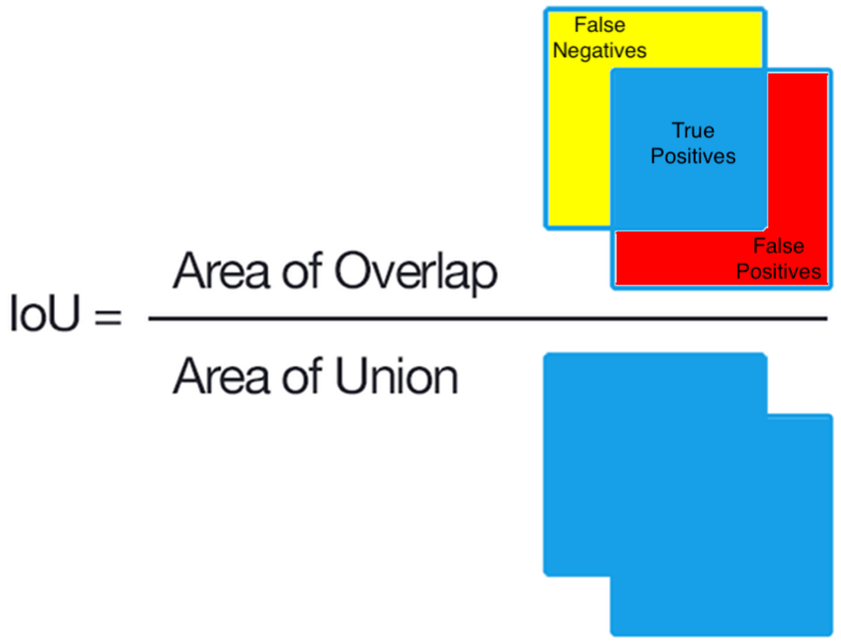


Figure 3.9. Demo of Intersection over union

True Positive(TP)

- IoU $>$ 0.5 that bounding box overlap with the ground truth box
- Correct classification (Predict Coke as Coke)

False Positive

- IoU $<$ 0.5 that bounding box overlap with the ground truth box
- Correct classification but duplicated bounding box

False Negative

- IoU > 0.5 that bounding box overlap with the ground truth box
- Wrong classification (Predict Coke as Pepsi)

To calculate the accuracy of detection of our object detector, we need to use true positives, false positive and false negative to show the performance of our detection system. In our case, we define accuracy based on calculating the number of bounding boxes labeled on the demo result compared with ground truth boxes. The equation is listed below.

$$Accuracy = \frac{\#\ TP\ bounding\ boxes}{\#\ total\ bounding\ box} \tag{3.11}$$

3.7.2 mAP

To calculate average precision(AP), we need to understand the function of precision and recall. Precision calculates the accuracy of the prediction, and recall function calculates the percentage of the prediction is been retrieved. The functions are listed as equation (3.12).

$$Precision = \frac{TP}{TP+FN}; Recall = \frac{TP}{TP+FN}; F1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{3.12}$$

According to rule of Pascal VOC detection evaluation from Russakovsky et al. (2014), the value of AP is calculated by a precision and recall curve from 0.0 to 1.0 with totally 11 points.

$$AP = \frac{1}{11} \sum\nolimits_{recall} Precision \cdot (Recall_i) \tag{3.13}$$

The average of the precision value forms the AP with a particular class. After all the class been evaluated, the average value of AP is called mean Average Precision(mAP). The mAP equation for a specific class C is shown 3.13. In notations,the mAP of Pascal VOC is ”mAP @ IoU=0.5” and mAP@0.5. In our case, we used VOC mAP rule to evaluate Faster-RCNN.

$$mAP = \frac{1}{|classes|} \sum_{c \in classes} \frac{\#TP(c)}{\#TP(c) + \#FP(c)} \tag{3.14}$$

There is another very popular public dataset Microsoft Common Objects in Context (MS-COCO) datasets published by T. Lin et al. (2014). In COCO's standard metric, instead of using IoU to set thresholds like VOC dataset rule, COCO AP is average over 10 different IoU thresholds from 0.5 to 0.95 to evaluate the mAP. To make it clear, AP for IoU 0.5 to 0.95 with 0.05 step size. Therefore, notation of mAP in COCO is "mAP@[0.5:0.95]". In our case, we utilized COCO mAP rule to evaluate YOLOv3 and CenterNet.

From T. Lin et al. (2014) COCO dataset mentioned, COCO mAP rule is a stricter metrics than Pascal VOC mAP. We expected to see mAP of VOC matric has a higher mAP than COCO metric.

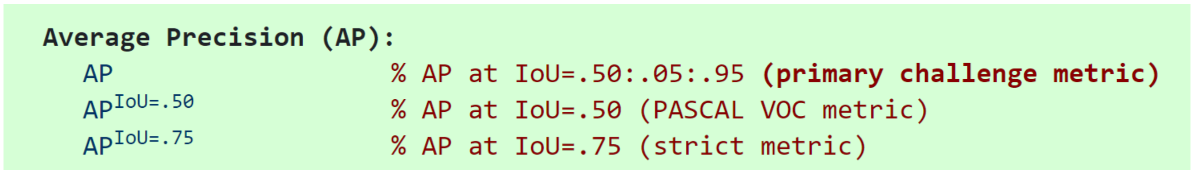

```
Average Precision (AP):
   AP                     % AP at IoU=.50:.05:.95 (primary challenge metric)
   AP^IoU=.50             % AP at IoU=.50 (PASCAL VOC metric)
   AP^IoU=.75             % AP at IoU=.75 (strict metric)
```

Figure 3.10. COCO mAP Metric

3.8 Summary

In summary, Chapter 4 provided the methodology of the thesis. The Bottle dataset has been demonstrated. We described three object detectors with their network architecture and algorithm. We also proposed algorithm and network architecture on the Categorical Auto encoder. At the end, we discussed evaluation criteria to validate the performance of each object detector on Bottle Dataset.

CHAPTER 4. EXPERIMENT AND RESULT

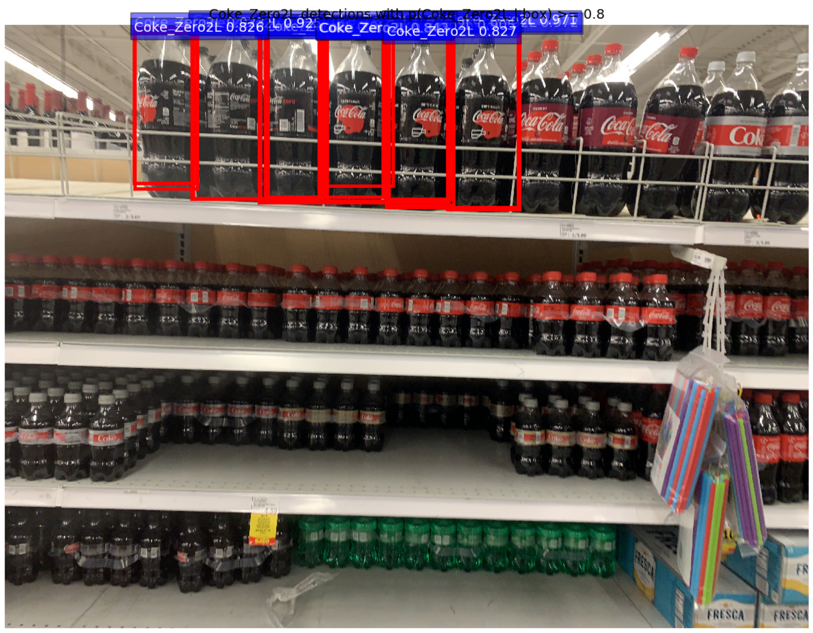
In total, we used three object detectors to train the model separately, and use the bottle dataset that divided into training, validation and testing set. In order to avoid bias, we randomized the Bottle dataset into training set, validation set, and test set that used on three different detectors. In training, we used the batch-size 4 (4 images) and trained Yolov3, Faster-RCNN, and CenterNet with a learning rate start at 0.001 for 1500 epochs. It reduced to 0.0001 or less within last 200 epochs depend on the detector. The training time depended on the difference of the detectors. In experiment, we used GPU of Nvidia GEFORCE GTX 1070 with 8GB on Yolov3 and Nvidia GEFORCE RTX 2060 with 6GB on Faster-Rcnn and CenterNet. The overall benchmark performance of two graphic cards is similar. RTX 2060 has around 10% better than GTX 1070.

During the training stage, we set the training set as input images with a fixed resolution to 512×512 on Faster-RCNN and CenterNet. The default resolution of YOLOv3 is 416×416, we did not change it during the training phase. Faster-RCNN took about 40 hours to train Bottle dataset (colored images) with one GPU Nvidia GEFORCE RTX 2060. CenterNet use ResNet-101 as network architectures took 15 hours to train bottle dataset (color and grayscale images) with one GPU Nvidia GEFORCE RTX 2060. CenterNet use ResNet-18 as network architecture only took 6 hours to train bottle dataset (color and grayscale images) with one GPU Nvidia GEFORCE RTX 2060. YOLOv3 took about 54 hours to train Bottle dataset (color and grayscale images) with GPU Nvidia GEFORCE GTX 1070.

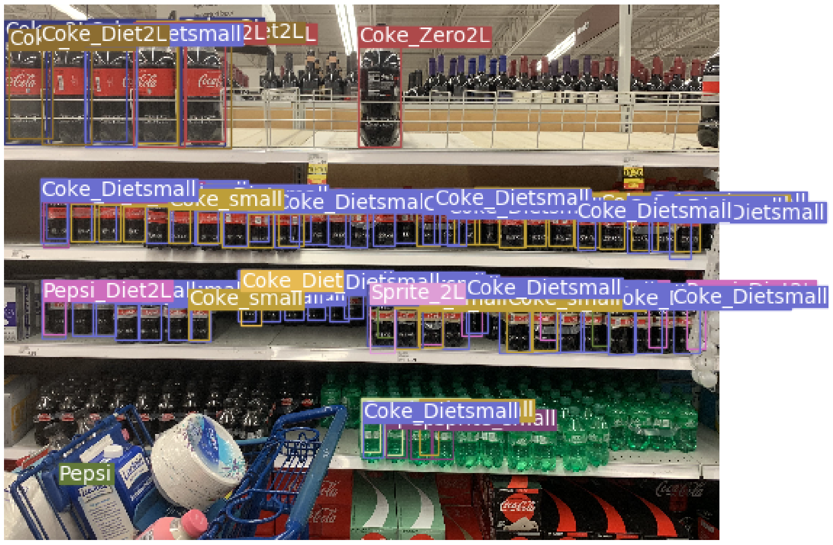
4.1 Object Detection Overlap

In our Bottle dataset, we contain images with shelves faced in a different angle. In our detection case, we need our detector able to detect the most front bottle displayed on the images. In normal, one object should only detect with one bounding box labeled with the correct classification. However, there are situations that detectors may detect one object into multiple bounding box and received the wrong classification.To solve the overlap bounding boxes, we increased the IoU threshold for NMS to decrease it to 0.01 will prevent an object from being detected if it overlaps with many bounding boxes. We changed the structure in Non-Maximum Suppression and the value of NMS-threshold.

- The first situation shows an object, which has been detected with multiple bounding boxes and the corrected classification.
- The second situation shows an object, which has been detected with multiple bounding boxes and incorrected classification.



(a) First Situation



(b) Second Situation

4.2 Grey Image

The research also discussed if grayscale images affected performance in detecting the Bottle dataset. When object detector learning from the feature map generated from the network that color is necessary information to detect an object. In our case, a colored image in Bottle dataset is a RGB image that contains geometry information of an object and color information. Compared to a RGB image, a grayscaled image only has a single dimension channel that derived from the RGB image. Therefore, a greyscaled image in Bottle dataset contains geometric information that lack of color information.

In experiment, we converted our 282 RGB images to 282 greyscale images. Thus, we used a traditional RGB to grayscale conversion formula and implement with OpenCV.

$$B' = 0.299R + 0.587G + 0.114B \tag{4.1}$$

(c) RGB to Grayscale

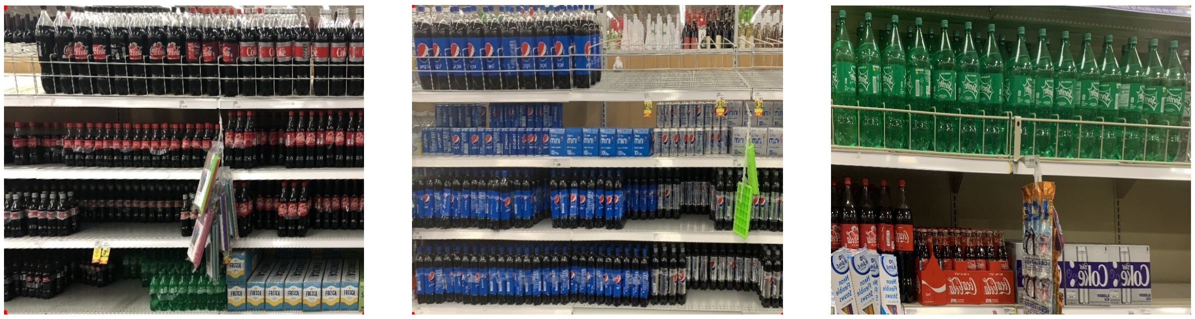
4.3 Perspective Transformation

The original test images contain some strongly skewed angled images. CenterNet not able to detect objects on those images very well. Since the object on the strongly skewed angled images can only see partial body of the bottle. In our case, we replaced strongly skewed images to the images that transferred with perspective transformation in the testing set. Figure (d) shows the example of strongly skewed angled images. we used perspective transformation to solve the issue with strongly skewed angled images can only view partial body of the bottle. It is a function that tracks four corners points on the image to warp it. Figure (e) shows the example of angled images after perspective transformation.

The perspective transformation projects from an original image to a new projective plane. The equation 3.1 showed the matrix calculation of the coordinates. $[x_1, y_1]^T$ represents the coordinates in the original image, and $[x, y]^T$ represents the coordinates in projective mapping. $\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}$ is the matrix of linear transformation including shearing,rotation, and scaling. Due to linear transformation, the left side of bottom is set to zero. Matrix 4.1 represents the general projective transform.

$$\begin{bmatrix} x_2 \\ y_2 \\ w \end{bmatrix} = \begin{bmatrix} a_1 & a_2 & t_x \\ a_3 & a_4 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \quad x = \frac{x_2}{w} \quad y = \frac{y_2}{w} \tag{4.2}$$

(d) Strongly skewed images

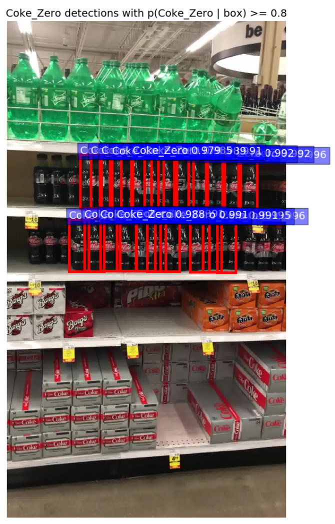

(e) After Perspective Transformation

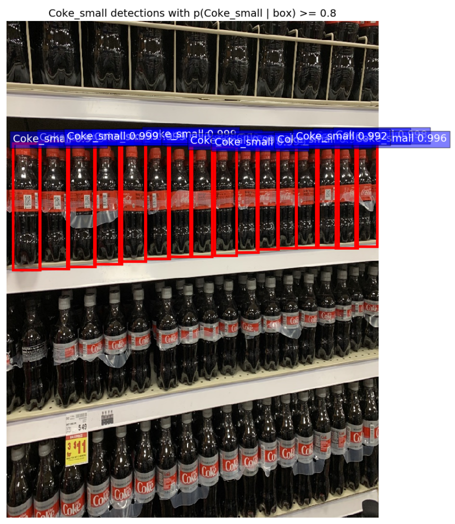
4.4 Results on Demo Images

In this section, we integrated Fast-RCNN, YOLOv3, and CenterNet to show the result of our detection on the testing set of the Bottle dataset. In our experiment, We compared performance on the Bottle Dataset with many different circumstances of object detectors. From the results on demo images, the baseline detectors could provide decent accuracy on detecting the objects on the shelf. But still obtained incorrect classification and position of the bounding box showed on the image. From the demo result, CenterNet displayed images achieve better performance than baseline detectors. To demo result from each detector, we selected the best model of each detector to demonstrate the best performance on detecting the bottles.

4.4.1 Faster-Rcnn

The result demo of Faster-Rcnn is listed below:





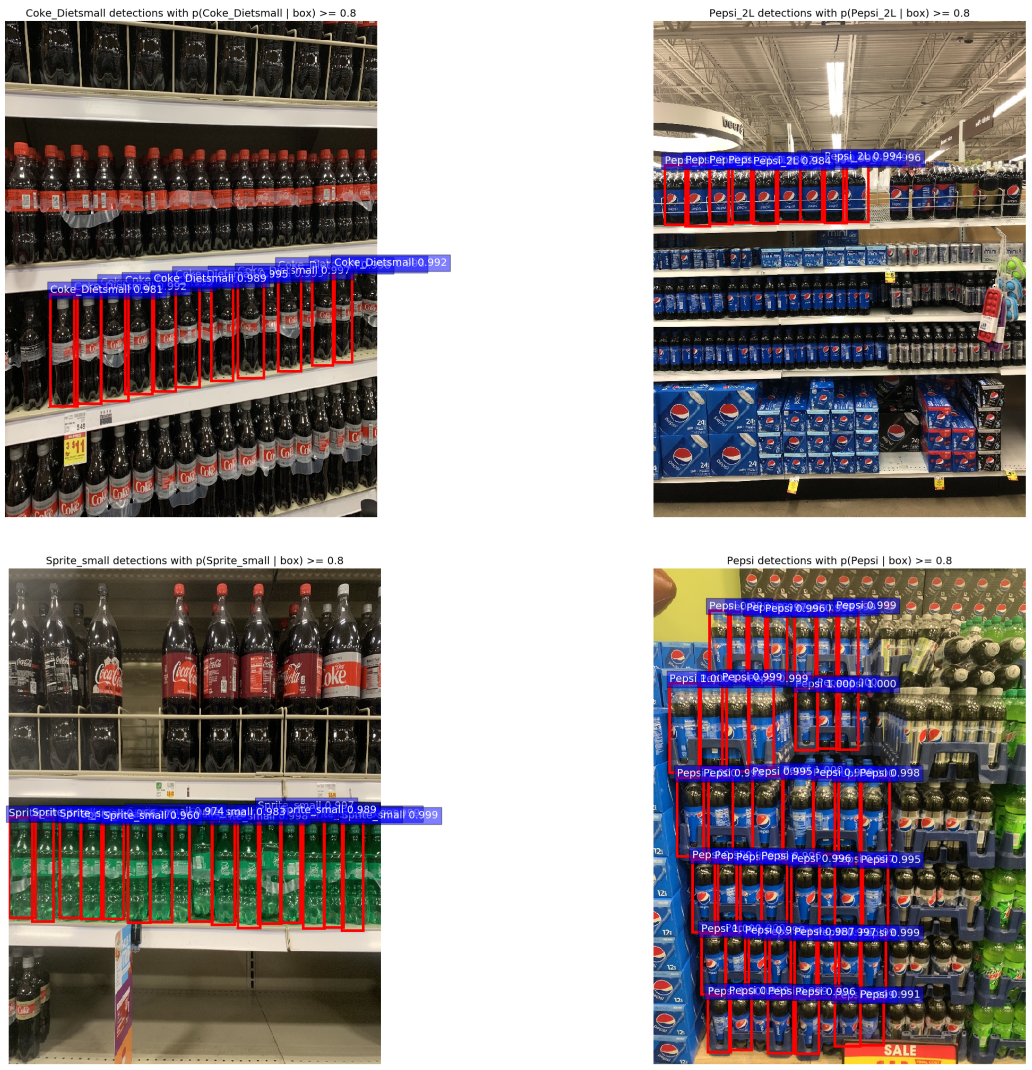

Coke_Dietsmall detections with p(Coke_Dietsmall | box) >= 0.8
Pepsi_2L detections with p(Pepsi_2L | box) >= 0.8
Sprite_small detections with p(Sprite_small | box) >= 0.8
Pepsi detections with p(Pepsi | box) >= 0.8

Figure 4.1. Demo Results on Faster-RCNN.

Figure 4.1 showed demo results on Faster-RCNN. The output images displayed Faster-RCNN has decent accuracy on detecting the objects on the shelves. But still obtained the issue with miss labeled objects and wrong position of bounding boxes.

4.4.2 Yolov3

The result demo of Yolov3 is listed below:

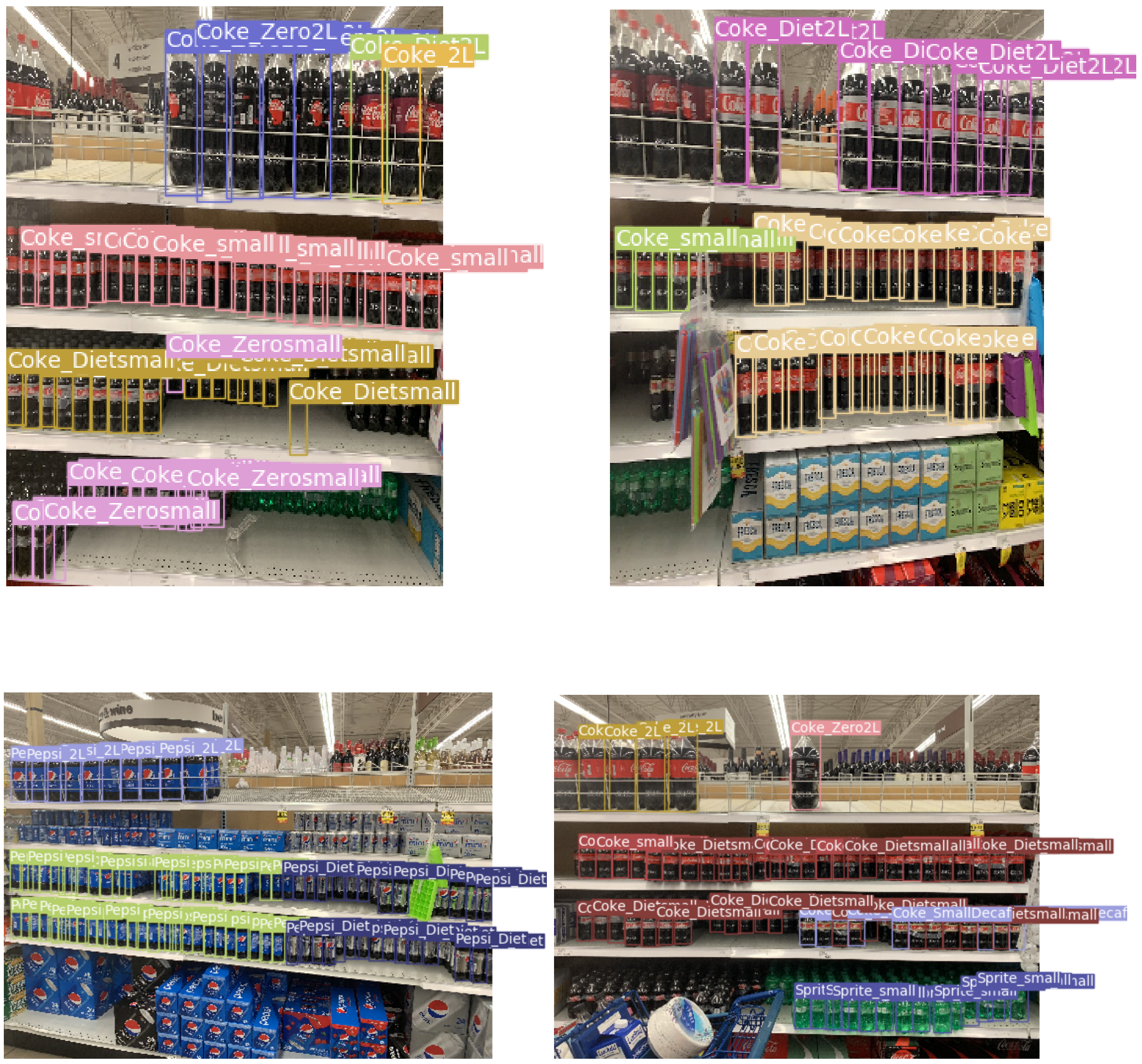
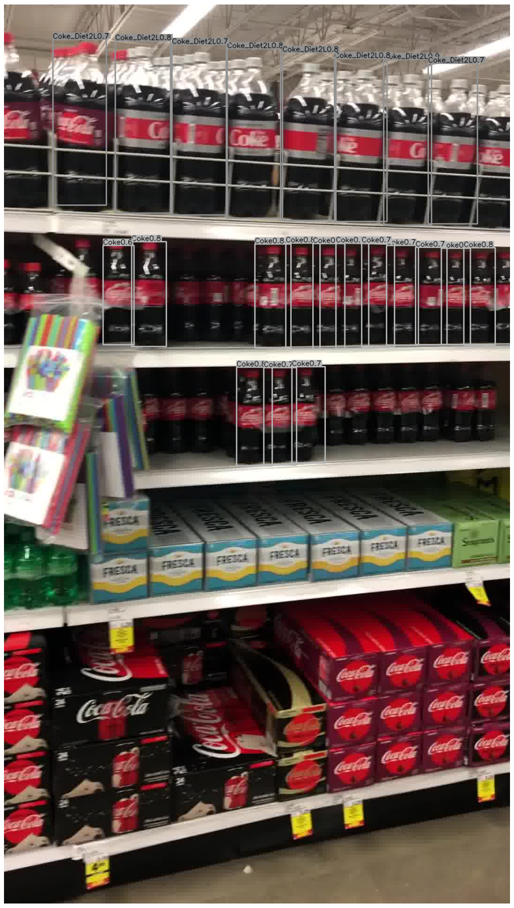


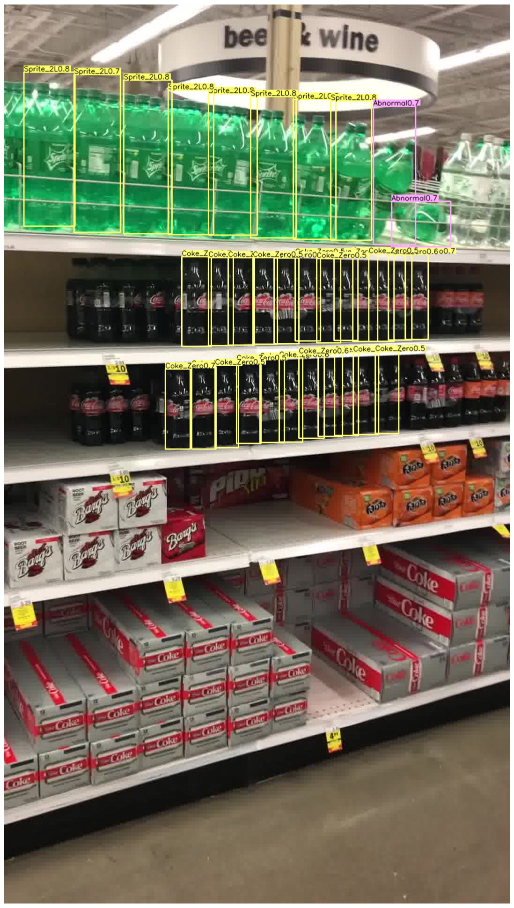
Figure 4.2. Demo Results on YOLOv3.

Figure 4.2 displayed demo results on YOLOv3. The output demo images of YOLOv3 have achieved decent accuracy on detecting the bottle on the shelves. However, there are still some problems with the result of YOLOv3 shown above. Most of the categories in the Bottle dataset had been identified correctly. We could still notice incorrect classification and miss labeled objects on the demonstration images.

4.4.3 CenterNet

The result demo of CenterNet is listed below:





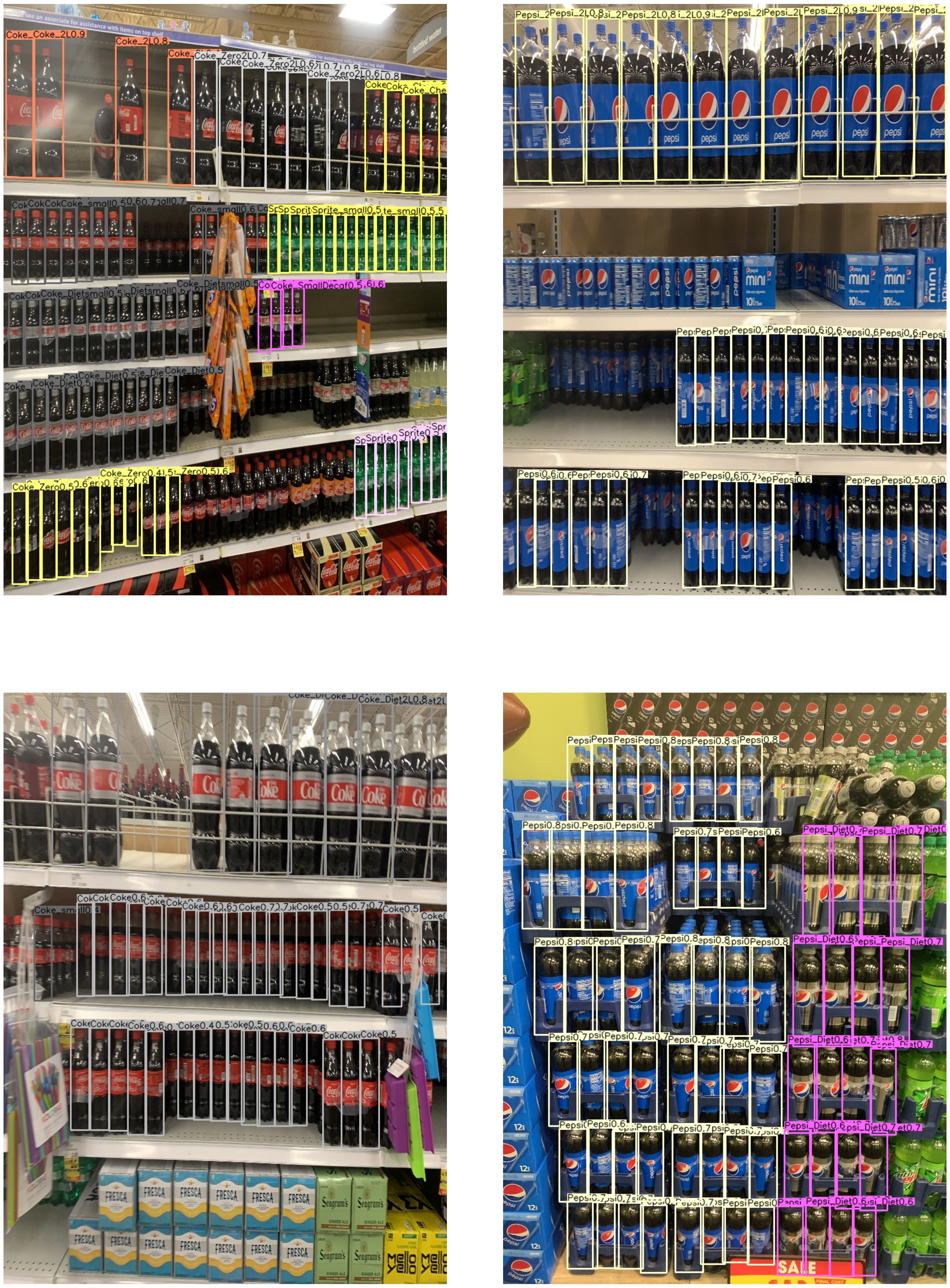
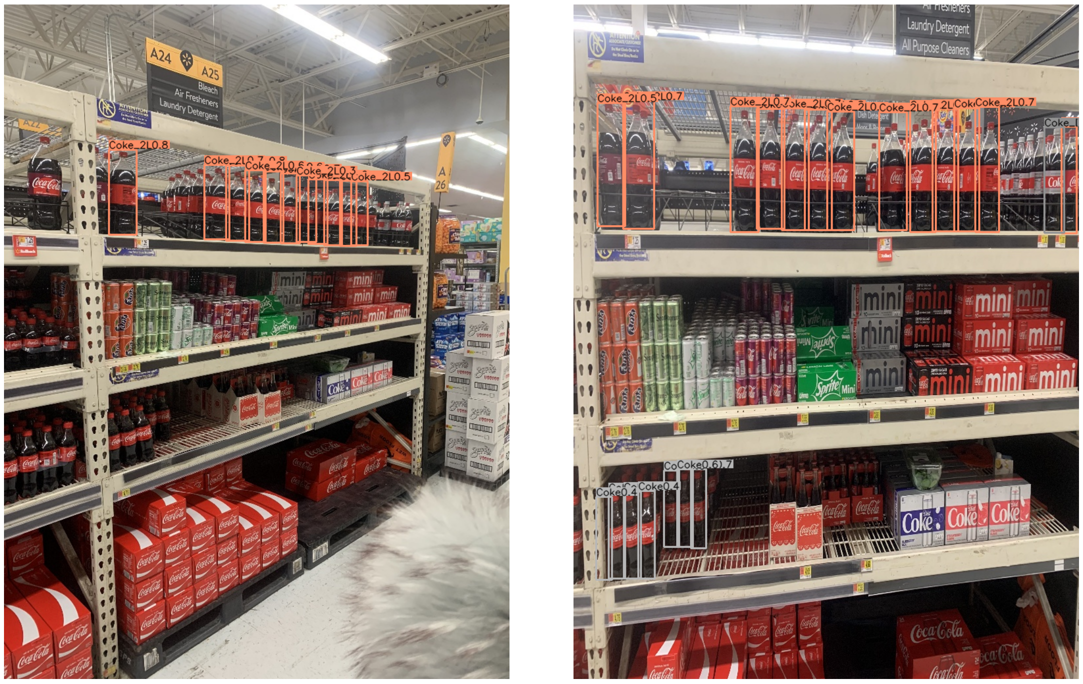


Figure 4.3. Demo Results on CenterNet.

Figure 4.3 displayed the detection result of CenterNet on images of the testing set. The output of demo images on CenterNet achieved the best accuracy that outperforms YOLOv3 and Faster-RCNN. Most of the ground truth bounding boxes in the Bottle dataset had been detected out with corrected classification. The category ”Abnormal” has been detected in the first image. Categories with different kinds of Pepsi, Coke, Diet Coke and Sprite had been clearly detected. We did not notice the incorrect classification and miss labeled objects on the demonstration images.

4.4.4 CenterNet with Perspective Transformation



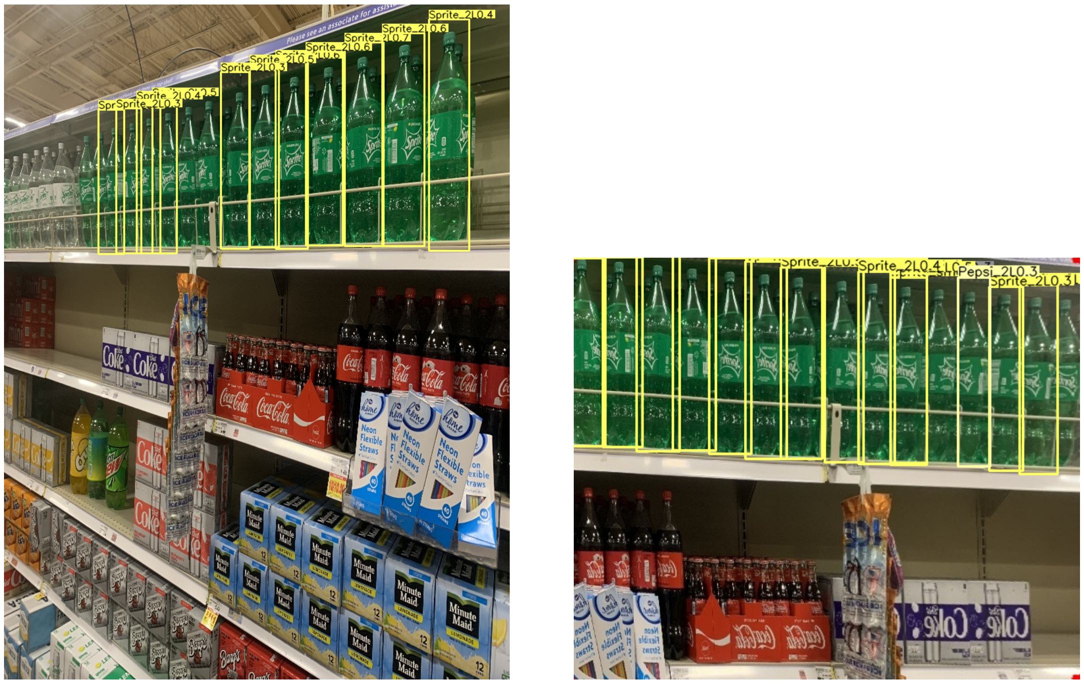


Figure 4.4. Results of perspective transformation.

Figure 4.4 displayed the detection result of CenterNet implemented with perspective transformation on the testing set. The left side images have shown the detection result on the original skewed images, with the fact that we can only see the partial body of the bottles for those strongly skewed images. The right side images have shown the detection result on images implemented with perspective transformation, which showed each bottle has been labeled with the correct category. The output results of perspective transformation proved that it can improve the detection performance on CenterNet.

4.5 Evaluation Criterion

In our thesis, we use Pascal VOC metrics to evaluate and COCO metrics to evaluate CenterNet, Faster-RCNN ,and YOLOv3.

Table 4.1 represent the CenterNet performance comparison on Bottle Dataset. Top: Hourglass network architecture; middle: ResNet-18 network architecture; bottom: ResNet-101 network architecture. Frame-per-second (FPS) measured on the same machine environment that means how many images the model can test in second. Colored means the colored images, and grayscale means the grayscaled images in the Bottle dataset. mAP@[0.5:0.95] is COCO's standard mAP metric that calculate average over 10 different IoU thresholds from 0.5 to 0.95. AP^{50} set as IoU threshold is 0.5 and AP^{75} set as IoU threshold is 0.75. PT means perspective transformation. In our case, we replaced strongly skewed images to the images implemented with perspective transformation in the testing set.

Table 4.1. *CenterNet performance on the Bottle Dataset*

Backbone	Training set	Test set	FPS	mAP@0.5.0.95	AP_{50}	AP_{75}	mAP_{+}
Hourglass	Colored	Colored	3.1	37.8	65.5	38.9	-
ResNet-18	Greyscale	Colored	112	49.9	81.7	63.5	-
ResNet-18	Colored+Greyscale	Colored	112	51.6	82.7	67.5	1.7
ResNet-18	Colored+Greyscale	Colored+PT	112	52.0	83.9	66.0	0.4
ResNet-18	Colored	Colored	112	53.7	84.4	**68.1**	1.7
ResNet-18	Colored	Colored+PT	**112**	**53.9**	**84.5**	67.7	0.2
ResNet-101	Greyscale	Colored	32	51.1	80.8	59.3	-
ResNet-101	Colored+Greyscale	Colored	32	55.0	84.6	69.3	3.9
ResNet-101	Colored+Greyscale	Colored+PT	32	55.2	84.9	69.4	0.2
ResNet-101	Colored	Colored	32	56.8	85.2	**69.7**	1.6
ResNet-101	Colored	Colored+PT	**32**	**57.2**	**85.4**	68.3	0.4

Faster-RCNN/YOLOv3 performance on the Bottle Dataset. Top: Faster-RCNN with VGG-16 as network architecture; bottom: YOLOv3 with Darknet-53 as network architecture. Frame-per-second (FPS) measured on the same machine environment that means how many images the model can test in second. Colored means the colored images, and grayscale means the grayscaled images in the Bottle dataset. mAP is the metric of Pascal VOC that refer as mAP @ IoU=0.5 and mAP@0.5.

Table 4.2. *Faster-RCNN/YOLOv3 performance on the Bottle Dataset*

	Dataset	FPS	mAP@0.5
Faster-RCNN	Colored images	0.9	58.92
YOLOv3	Greyscale	28.5	70.58
YOLOv3	Colored	28.5	75.54
YOLOv3	Colored+Greyscale	**28.5**	**76.50**

In our thesis, we use Pascal VOC metrics and COCO metrics to evaluate CenterNet, Faster-RCNN,and YOLOv3. We compare CenterNet performance on the Bottle Dataset with different circumstance in Table 4.1, we compared colored images, grayscaled images, and perspective transformation on CenterNet. The result shows that CenterNet with ResNet-18(colored) achieves an mAP of 53.7% and AP_{50} of 84.4% at 112 FPS.After replaced skewed images to the PT images in the testing set. For ResNet-18(Colored+PT) the mAP improves from 53.7% to 53.9% and AP_{50} from 84.4 to 84.5% at 112 FPS. CenterNet with ResNet-101(Colored) achieves an mAP of 56.8% and AP_{50} of 85.2% at 32 FPS. After replaced skewed images to the PT images in the testing set. For ResNet-101(Colored+PT) the mAP improves from 56.8% to 57.2% and AP_{50} from 85.2% to 85.4% at 32 FPS. From the result given above, We can conclude that implemented perspective transformation on strongly angled images can improve the mAP. ResNet-101 has a better mAP result overall compare to ResNet-18. The model from ResNet-101(Colored+PT) achieves the best result of mAP and AP_{50} of CenterNet.

We compared the mAP performance of Faster-RCNN and YOLOv3 on the Bottle dataset by using Pascal VOC metrics in Table 4.2. The result shows that Faster-RCNN achieves an mAP of 58.92% at FPS 0.9. YOLOv3(colored+greyscaled) achieves an mAP of 76.50 at FPS 28. The mAP of Faster-RCNN and YOLOv3 seems better than CenterNet, but COCO mAP is a much stricter metrics compare to Pascal VOC mAP. Pascal VOC as mAP@IoU=0.5 can refer as COCO AP_{50}. Thus, CenterNet outperforms YOLOv3 and Faster-RCNN. From the FPS showed above, CenterNet with ResNet-18 is way faster than ResNet-101 and YOLOv3.

To discuss the importance and contribution of the color information for detecting the objects. We can compare the performance of each network architectures for training with greyscaled images and greyscaled combined with colored images. The result shown ResNet-18(greyscaled) achieves an mAP of 49.9% and AP_{50} of 81.7% at 112 FPS. After add colored images into the training and validating set. ResNet-18(Colored+greyscaled) the mAP improves from 49.9% to 53.9% and AP_{50} from 81.7 to 84.5%. The learning of colored information help improve the performance of mAP up to 4%. ResNet-101(greyscaled) achieves an mAP of 51.1% and AP_{50} of 80.8% at 32 FPS. After add colored images into the training and validating set. ResNet-18(Colored+greyscaled) the mAP improves from 51.1% to 55.0% and AP_{50} from 80.8 to 84.6%. The learning of colored information help improve the performance of mAP up to 3.9%.

To calculate the accuracy of detection on CenterNet, we define accuracy based on calculation of the number on bounding boxes that has been labeled (IoU>0.5) on the demo result compared with ground truth boxes. We choose to test the accuracy on model ResNet-101(Colored+PT). Because it outperform other circumstances.The equation is listed at 4.3. In our case, we calculated 1845 total bounding boxes with 1754 total number of TP bounding boxes that have been selected from 56 images as our test set. The result of accuracy of detection on CenterNet is 95.06%, It surpass presumed requirement accuracy of 93% on detecting the objects.

$$Accuracy = \frac{\#\ TP\ bounding\ boxes}{\#\ total\ bounding\ box} = \frac{1754}{1845} = 95.06\% \tag{4.3}$$

4.6 Summary

The chapter provided the details in the experiment with three object detectors in the Bottle dataset and the results of the thesis.

CHAPTER 5. CONCLUSION AND FUTURE WORKS

5.1 Conclusion

In the thesis dissertation, we presented three different object detectors to solve the detection issue on dense objects. From the result of my experiments,Faster-RCNN and YOLOv3 as our baseline detectors provide a decent performance on detecting the objects on the shelves. On CenterNet, We compared performance on the Bottle Dataset with many different circumstances. The ResNet-101(colored+PT) achieves the best result of mAP and AP_{50} of CenterNet that outperform other network architecture. The ResNet-18(colored+PT) has the fastest model that achieves a reputable performance of 53.9% COCO mAP metric at 112 FPS. Which is triple as fast as ResNet with decent accuracy. Moreover, we proved perspective transformation can be implemented on CenterNet to improve its performance that increases the mAP@[0.5.0.95]. Perspective transformation solves the issue when the detector does not achieve a good result on strongly angled images. We experimented perspective transformation on CenterNet, For ResNet-101(Colored+PT) the mAP improved from 56.8% to 57.2% and AP_{50} from 85.2% to 85.4%. It improved the performance on ResNet-18 as well.To discuss the importance and contribution of the color information for detecting the objects on the shelf. The result from colored images contains the geometric information and color information, while greyscaled images only contain geometric information. The additional color information helped ResNet-18 improved its mAP from 49.9% to 53.9% and ResNet-101 improved its mAP from 51.1% to 55.0% that both up to 4%.We concluded that colored information does contribute to the performance in detecting the objects on the shelf, but it does not contribute as much as geometric information provided for learning its information.

5.2 Future Works

Due to the limit of time, we can improve our thesis with many following ways:

- We will rewrite the Pascal VOC mAP@50 to COCO mAP[0.5.0.95] for Faster-RCNN and YOLOv3.
- we could add more images into the current dataset and published for public research. The current dataset is small but aimed at a specific environment.
- We could build in perspective transformation into the training phase as data augmentation.

REFERENCES

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., & Tian, Q. (2019). Centernet: Keypoint triplets for object detection. *CoRR*, *abs/1904.08189*. Retrieved from `http://arxiv.org/abs/1904.08189`

Eggert, C., Zecha, D., Brehm, S., & Lienhart, R. (2017). Improving small object proposals for company logo detection. *CoRR*, *abs/1704.08881*. Retrieved from `http://arxiv.org/abs/1704.08881`

Girshick, R. B. (2015). Fast R-CNN. *CoRR*, *abs/1504.08083*. Retrieved from `http://arxiv.org/abs/1504.08083`

Girshick, R. B., Donahue, J., Darrell, T., & Malik, J. (2013). Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, *abs/1311.2524*. Retrieved from `http://arxiv.org/abs/1311.2524`

Hearst, M. A. (1998, July). Support vector machines. *IEEE Intelligent Systems*, *13*(4), 18–28. Retrieved from `https://doi.org/10.1109/5254.708428` doi: 10.1109/5254.708428

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd international conference on international conference on machine learning - volume 37* (pp. 448–456). JMLR.org. Retrieved from `http://dl.acm.org/citation.cfm?id=3045118.3045167`

Law, H., & Deng, J. (2018a). Cornernet: Detecting objects as paired keypoints. *CoRR*, *abs/1808.01244*. Retrieved from `http://arxiv.org/abs/1808.01244`

Law, H., & Deng, J. (2018b). Cornernet: Detecting objects as paired keypoints. *CoRR*, *abs/1808.01244*. Retrieved from `http://arxiv.org/abs/1808.01244`

LeCun, Y., Jackel, L., Bottou, L., Brunot, A., Cortes, C., Denker, J., ... others (1995). Comparison of learning algorithms for handwritten digit recognition. In *International conference on artificial neural networks* (Vol. 60, pp. 53–60).

Lin, G., Milan, A., Shen, C., & Reid, I. D. (2016). Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *CoRR*, *abs/1611.06612*. Retrieved from `http://arxiv.org/abs/1611.06612`

Lin, T., Goyal, P., Girshick, R. B., He, K., & Dollár, P. (2017). Focal loss for dense object detection. *CoRR*, *abs/1708.02002*. Retrieved from `http://arxiv.org/abs/1708.02002`

Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., … Zitnick, C. L. (2014). Microsoft COCO: common objects in context. *CoRR*, *abs/1405.0312*. Retrieved from `http://arxiv.org/abs/1405.0312`

Merler, M., Galleguillos, C., & Belongie, S. (2007, June). Recognizing groceries in situ using in vitro training data. In *2007 ieee conference on computer vision and pattern recognition* (p. 1-8). doi: 10.1109/CVPR.2007.383486

Newell, A., Yang, K., & Deng, J. (2016). Stacked hourglass networks for human pose estimation. *CoRR*, *abs/1603.06937*. Retrieved from `http://arxiv.org/abs/1603.06937`

Redmon, J., Divvala, S. K., Girshick, R. B., & Farhadi, A. (2015). You only look once: Unified, real-time object detection. *CoRR*, *abs/1506.02640*. Retrieved from `http://arxiv.org/abs/1506.02640`

Redmon, J., & Farhadi, A. (2016). YOLO9000: better, faster, stronger. *CoRR*, *abs/1612.08242*. Retrieved from `http://arxiv.org/abs/1612.08242`

Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. *CoRR*, *abs/1804.02767*. Retrieved from `http://arxiv.org/abs/1804.02767`

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., … Li, F. (2014). Imagenet large scale visual recognition challenge. *CoRR*, *abs/1409.0575*. Retrieved from `http://arxiv.org/abs/1409.0575`

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., … Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, *115*(3), 211-252. doi: 10.1007/s11263-015-0816-y

Shmelkov, K., Schmid, C., & Alahari, K. (2017). Incremental learning of object detectors without catastrophic forgetting. *CoRR*, *abs/1708.06977*. Retrieved from `http://arxiv.org/abs/1708.06977`

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., … Rabinovich, A. (2014). Going deeper with convolutions. *CoRR*, *abs/1409.4842*. Retrieved from `http://arxiv.org/abs/1409.4842`

Tonioni, A., Serra, E., & di Stefano, L. (2018). A deep learning pipeline for product recognition on store shelves. *CoRR*, *abs/1810.01733*. Retrieved from `http://arxiv.org/abs/1810.01733`

Varadarajan, S., Kant, S., & Srivastava, M. M. (2019). *Benchmark for generic product detection: A low data baseline for dense object detection.*

Varadarajan, S., & Srivastava, M. M. (2018). Weakly supervised object localization on grocery shelves using simple FCN and synthetic dataset. *CoRR*, *abs/1803.06813*. Retrieved from `http://arxiv.org/abs/1803.06813`

Varol, G., & Kuzu, R. S. (2015). Toward retail product recognition on grocery shelves. , *9443*, 46 – 52. Retrieved from `https://doi.org/10.1117/12.2179127` doi: 10.1117/12.2179127

Wang, K., Fang, B., Qian, J., Yang, S., Zhou, X., & Zhou, J. (2019, 12). Perspective transformation data augmentation for object detection. *IEEE Access*, *PP*, 1-1. doi: 10.1109/ACCESS.2019.2962572

Zhao, H., Gallo, O., Frosio, I., & Kautz, J. (2015). Loss functions for neural networks for image processing. *CoRR*, *abs/1511.08861*. Retrieved from `http://arxiv.org/abs/1511.08861`

Zhou, X. (2018). Transferring scale-independent features to support multi-scale object recognition with deep convolutional neural network. In *Proceedings of the 26th acm sigspatial international conference on advances in geographic information systems* (pp. 614–615). New York, NY, USA: ACM. Retrieved from `http://doi.acm.org/10.1145/3274895.3282797` doi: 10.1145/3274895.3282797

Zhou, X., Wang, D., & Krähenbühl, P. (2019). Objects as points. *CoRR*, *abs/1904.07850*. Retrieved from `http://arxiv.org/abs/1904.07850`