Naïve Approach for Bounding Box Annotation and Object Detection Towards Smart Retail Systems

Pubudu Ekanayake[1(✉)], Zhaoli Deng[1], Chenhui Yang[1(✉)], Xin Hong[2(✉)], and Jang Yang[3]

[1] Computer Science Department, School of Information Science and Engineering, Xiamen University, Xiamen 361005, China
pubudu1ekanayake@gmail.com, 304628356@qq.com, chyang@xmu.edu.cn
[2] College of Computer Science and Technology, Huaqiao University, Xiamen 361021, China
xinhong@hqu.edu.cn,10409035@qq.com
[3] Cognitive Science Department, Sixth College, University of California, San Diego, CA 92092, USA

Abstract. It is becoming a trend that companies use smart retail stores to reduce the selling cost, by using the sensor technologies. Deep convolutional neural network models which are pre-rained for the Object detection task achieve state-of-the-art result in many benchmark. However, when applying these algorithms to the intelligent retail system to help automated checkout, we need to reduce the manual labelling cost of making retail data sets, and to achieve real-time demand while ensuring accuracy. In our paper, we propose a naive approach to get first portion of the bounding box annotations for a given custom image dataset in order to reduce manual cost. Experimental results show that our approach helps to label the first set of images in short time of period. Further, the custom module we designed helped to reduce the number of parameters by 41.77% for the YOLO model maintaining the original model's accuracy (85.8 mAP).

Keywords: Smart retail system · Object detection · Bounding box annotation · Convolutional neural network · YOLO

1 Introduction

Some tech companies such as Amazon and Orange do researches and developments of smart retail stores with combining with the cutting edge technologies to reduce the selling cost. These newest technologies include the sensor technology, computer vision technology, AI technology and much more as either stand alone or a combination. Unlike in traditional retail stores, in smart retail stores the store automatically identifies and calculate the price for the products in a buying process. By implementing the smart retail stores, they help to reduce

G. Wang et al. (Eds.): SpaCCS 2019 Workshops, LNCS 11637, pp. 218–227, 2019.
https://doi.org/10.1007/978-3-030-24900-7_18

the requirement of human employees in the selling activities for a store. Further, the requirement of monitoring the products will also be reduced due to the monitoring capabilities of a smart retail store.

When considering the smart retail stores, much literature can be found which uses RFID technology [1]. When use RFID sensors we need to tag each object with the RFID sensors. However, with the development of the AI field and computer vision techniques, now most of the companies try to attach the innovations from the AI to smart retail stores. Paper [2] discusses about the technologies used in the Amazon go smart store. The research paper [3] has developed a smart checkout system using the object detection and classification algorithms. They have used YOLO [4,5] object detection algorithm and another trained classifier to identify the object category type. Further, much work can be found under the product detection and classification in the recent past years. Paper [6] tried to categorize thousands of fine-grained products with few training samples. Non-parametric probabilistic models were used for initial detections and then used CNNs where applicable. Paper [7] used active learning process to improve the recognitions in their models continuously. Papers [8–10] focus about recognizing and finding the misplaced products in the shelves using different techniques such as BoW techniques, classical feature extraction procedures (SIFT,HoG), DNN, etc. Image annotations, many approaches were taken to reduce the bounding box annotation cost in recent past years. Few are box verification series [11], point annotations [12,13] and eye tracking [14]. Among above the paper [11] has produces much quality detectors at low cost. Apart from above mentioned approaches human supervision methods also used for bounding box annotation [15–17].

Our designed solution includes a camera to capture the moving object/product in the smart store environment. It removes some of the issues of shelf monitoring methods had. In the coming phases, we will improve our project by applying motion sensors and heat sensors to improve the current results. Further, we will experiment with Reinforcement Learning methods by using RFID sensors to train models. Experimental results show that our designed custom module helped to reduce the number of parameters in the YOLO object detection model by 41.77% while maintaining the original accuracy.

2 System Design of Smart Retail Store

Our system (prototype) consists of a shelf equipped with image sensor, object detection model, product dataset and cloud server.

Our dataset is made up of product images. As shown in the red box in the Fig. 1, we make a training dataset based on our method. The original image dataset is not labelled, and our method used to annotate the bounding boxes for original images and generate the annotation files (.xml). The advantage of our method is that it can reduce the cost of manual labelling.

At the moment, Our prototype is only equipped with image sensors. As shown in the green box in the Fig. 1, Our sensor will capture images when customers

shop. These images will be passed as input to our object detector. Then the detector outputs the category type of the product and its coordinates on of the detected frame, and the output is further analyzed to obtain the user's purchase list. Our main contribution is to design a new module to reduce the computational complexity of the model. Most importantly we placed our sensors to avoid some complexities such as monitoring the shelf, counting the number of products inside the shelf, etc. However, in an indirect manner our solution helps to monitor the shelves.

As we know the packaging of the product is frequently updated. So our model also needs to be updated. However to avoid the problems occurred for updating the system regular basis will be handled by uploading the trained model to a cloud server. Hence, we can train the models for new data in the back-end and update the cloud server model without stopping the smart retail system in the front-end, which leads to encourage the small retail companies to grab the new technology.

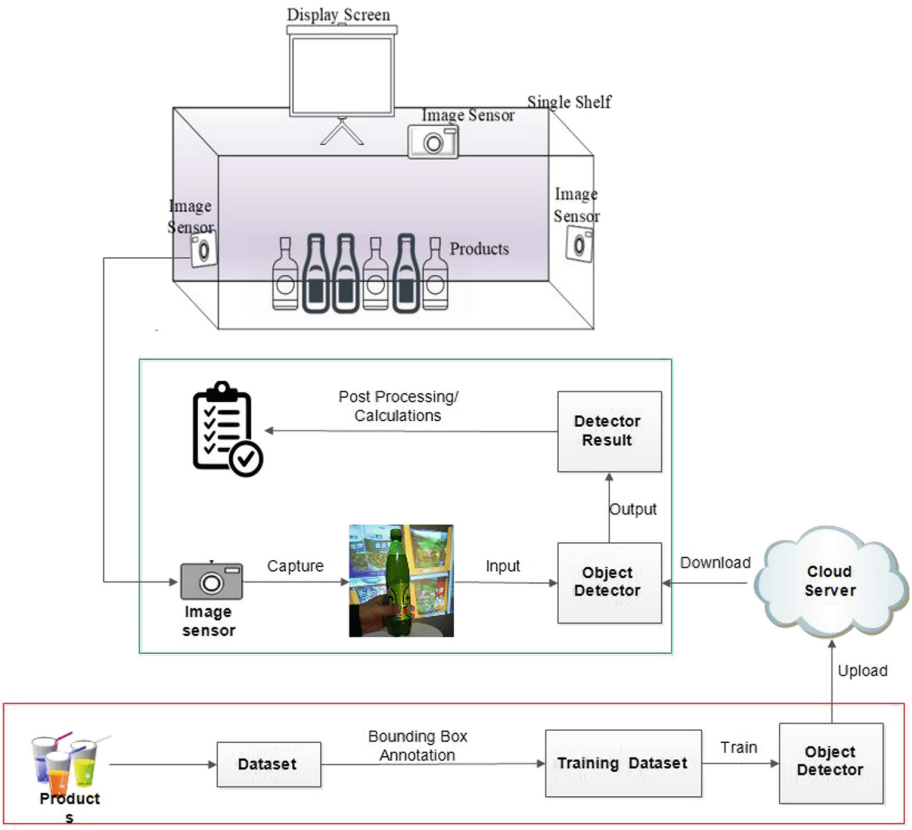

Fig. 1. Flow chart of the smart cambin.

3 Proposed Method

In this section, we will introduce our method and algorithm details.In addition,we will introduce the module we designed to reduce the computational complexity.

3.1 Naïve Bounding Box Annotation

Our approach requires a pre-training model, a feature extractor and trained a simple classifier. Its procedure is shown in Fig. 2. Image dataset which is required to be annotated will be fed into the pre-trained object detection model. From the pre-trained object detector we get the top-left and bottom-right corner coordinates of the detections. Then we crop image based on these coordinates. Next we extract feature vectors from cropped images by using the feature extractor (VGG16 [18]). Then we use simple classifiers to classify (we already trained a logistic regression classifier from very small number of cropped images) these features to get their classification categories. Finally it generates the annotation file (in .xml format) containing the coordinate information and the object category.

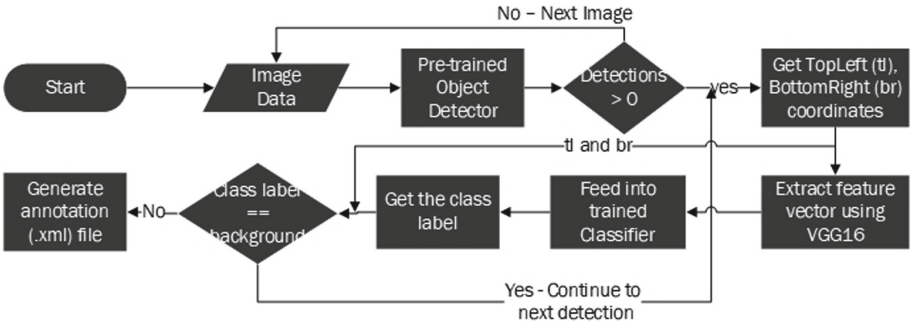

Fig. 2. Naïve bounding box annotation approach–flow chart.

The public pre-trained object detection model mentioned before is trained on large datasets such as COCO [19], iNaturalist Species Detection Dataset [20], etc. When training the simple classifiers, approximately 50 good-cropped images were selected for each category type and 446 cropped images as background including the hard negatives. 1266 total cropped images were used in training the simple feature classifier.

We extract feature maps for cropped images. Features are extracted for the images to train a simple classifiers (SVM/Logistic Regression). We use VGG16 [18] model (without the final fully connected layers) which is trained on "ImageNet" [21] dataset as the feature extractor. Pre-training model is used to extract feature maps with rich features, which our small dataset does not represent. We

used data augmentation method while extracting the feature vectors. Hence, we were able to increase the number of extracted feature samples for training the classifier that led to increase the accuracy of the classifier almost by 4.0% compared to the feature set, which used without data augmentation.

Manual annotation is reduced by this method. After the annotation, we have done a human supervision and incorrectly, annotated images and annotations (xml files) were removed. Using the annotated images, we can start training our object detector to annotate the rest of the images (similar to active learning approaches) and the procedure mentioned in [22].

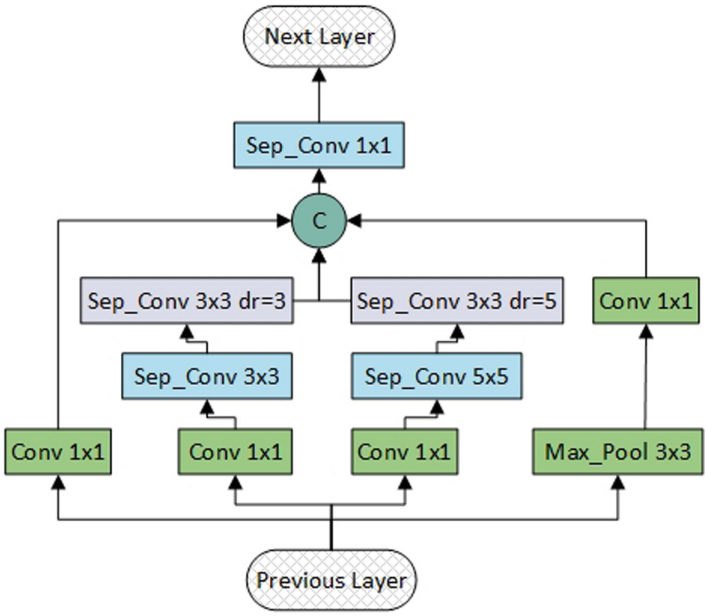

Fig. 3. Designed module for the proposed convolutional neural network architecture. dr = dilation rate.

3.2 Custom Module

Basically to design our proposed architecture we have used the YOLOV2 [4] (tiny-yolo version) algorithm. We have tried out few different architectures for the feature extraction part of the network while using the yolo algorithm itself (loss function). For the proposed module and the architecture from us, used the concept about depthwise convolutions followed by pointwise convolutions mentioned in the MobileNetV1 [23] and paper [24] to reduce the computational complexity of the model by decreasing the number of matrix multiplicative operations. Further, we modified the architecture to focus on nearby features aswell

using the dilated convolutions with normal convolutions inspired by the paper RFBNet [25]. However, to reduce the computational complexity we have followed the concept of Depthwise Convolutions from [25] for the dilated convolutions as well for designed module. Output number of activations were decided by considering the paper [26] and multiplying the values mentioned in the original paper by factor 1/2. We could not do any more experiments on fine-tuning those values. Hence modules designed (Fig. 3) by us is basically the combination of Depthwise Convolutions [23] and Dilated Convolutions [25] which used in a manner in the paper GoogleNet [26] with 2 branches.

Model Serving. Should be done in a cloud platform and while maintaining a model in the local machines when considering the smart retail system. Once the smart retail system is up and running we should perform model training and model updating in the backend without stopping the smart retail system. We happened to found that the best way to perform this task is using a cloud platform. Train the models, update the weight files in the backend, and update it in the cloud system. Therefore, without any interruptions and with very small delay we can update the local machines in the smart retail system.

4 Result

In this section, we show some results on the experiments conducted during working on the project.

Table 1. Accuracies for cropped image classifier – Naïve bounding box annotation

	Accuracy	
Classifier model	W/O augmentation	Augmentation
SVM	88.30%	92.00%
Logistic regression	88.41%	92.00%
CNN	-	92.91%

4.1 Simple Classifiers

We have trained Simple classifiers to classify the cropped images. As shown in the Table 1 there is a significant improvement of the accuracy when using the features extracted with the data augmentation. However, we have not noticed any significant different of training SVM model, Logistic Regression model and CNN model. For our project we choose the Logistic Regression model as the classifier. In generally we suggest it would be much easier to go with logistic regression model than other two models as the SVM took considerably large amount of time to train.

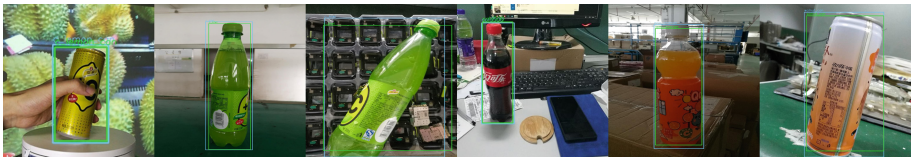

Fig. 4. Some image results obtained from Naïve bounding box annotation.

Naive Bounding Box Annotation. Figure 4 shows some of the results taken from the naive bounding box annotation experiments. As we can see that for the custom image datasets, the proposed naïve bounding box annotation method could be used and obtained some good results. Object localization was done using the pre-trained object detector, and classification was done using the trained logistic regression model. Sample results show, two bounding boxes in each image. One is the ground-truth bounding box, which was drawn by the humans manually. The other bounding box and the label was predicted by our naive bounding box annotation approach.

Table 2. Results for naive bounding box annotation

Model	Image dataset	Detections	(%)	Time(h)
mask_rcnn_inception_resnet	RPC [27]	14,921	27.76	8.6
mask_rcnn_inception_resnet	Custom dataset	734	56.38	0.3
faster_rcnn_resnet50_coco	Raccoon dataset	112	56.00	0.02

Table 2 depicts the results, if IoU (intersection over union) of the detections/predicted boxes and the ground truth boxes are equal or greater than 0.75. Last column represents the time taken to perform the bounding box annotations for the image datasets in hours. The results suggests that specially for a large image dataset we can use our naive image bounding box annotation approach to get the first set of bounding box annotations for a custom image dataset. We used only singe objects training images from RPC [27] image dataset. Since the RPC [27] image dataset has 53,739 training images in total.

4.2 Results for Some Experiments Conducted on Object Detection

Object detection was done using a video stream taken from single camera. As shown in the Table 3 the best mAP results are achieved by the Tiny YoloV2 [6] model. It has the second highest number of parameters and second slowest FPS speed among the experimented object detection models. For the real time applications mobilentV2 [23] is quite good with considerably low number of parameters and a better speed compared to the other models. The model we proposed has very low number of parameters and competitive mAP percentage

Table 3. Comparisons done within smaller one-stage object detectors, which our custom data set trained.

Model	Input layer size	mAP (%)	FPS	# of parameters (millions)
Tiny yolo v2	416×416	85.80	2.02	15.8
SSDMobilenetV2	300×300	68.04	7.06	(approximately) 3.4
SSD (VGG16 backbone)	300×300	68.31	0.80	25.7
MobilenetV2 (as backend)	301×301	67.02	3.56	3.4
Ours	416×416	77.47	2.16	1.2
Ours	301×301	69.84	3.93	1.2

Table 4. Effectiveness of the custom module.

Model	Input layer size	mAP (%)	FPS	# of parameters (millions)
Tiny yolo v2	416×416	85.80	2.02	15.8
Tiny yolo v2 – replaced module	416×416	86.32	2.41	9.2

to SSDMobilenetV2 model architecture and having much faster frame rate compared to Tiny Yolo V2 [6]. Comparing our model which has 416×416 input layer size and the Tiny YoloV2 model, our model got low accuracy while passing the FPS speed and maintaining very low number of learning parameters (almost 90% reduction) which indicates it required very small storage capacity. Hence, we hope this model architecture can be used in mobile applications. We hope the FPS speed can be increased by replacing the 5×5 Seperable Convolutions in the early phase of our designed network with the 3×3 seperable convolutions.

Original tiny yolo uses size 416 for the input image size while SSD MobileNet models used 300 and our model used 301 as input image size. We believe that input image size and the higher number of parameters lead to achieve higher number of mAP for the Tiny Yolo V2 model. MobileNetV2 architecture design used only 3×3 kernel sizes and 1×1 kernel sizes.

Effectiveness of the Custom Module. We performed a simple experiment to check the effectiveness of the designed custom module. We replaced last second and third to the last layers of the original Tiny-YOLO model with the designed custom module. Comparison results with the original Tiny-YOLO model was given in the Table 4. We have used the custom image dataset for this experiment. According to the results, we were able to reduce the number of parameters of the model by 41.77% while maintaining the original result with a slight increase of the FPS speed.

5 Conclusion

We have proposed a system architecture for a smart retail system in this paper and further made 3 major contributions in this paper: (1) propose a naïve approach to get first portion of the bounding box annotations for a given custom image dataset. (2) A shallower and lightweight network architecture. (3) Novel module design considering the receptive fields of convolutions more at the end of the architecture. While discussing above major facts we describe the application of Convolutional Neural Networks and Object tracking to build a Smart Retail store. Using our proposed naïve bounding box annotation method, it reduces largely manual work, which has to be done.

We have used a simple camera to capture the environment and performed object detection. Further, we have used another camera to detect the face of a customer which we have not discussed in detail in this paper. In future work, the trained model will be further tested by hosting in a cloud server and add the data mining section to suggest products using the age and gender predictions while continue to improve the accuracy on both models for object detection and face, gender classification. Further, we will use motion sensors, more cameras to localize the moving object which will help to improve the accuracy of the smart retail store.

References

1. Domdouzis, K., Kumar, B., Anumba, C.: Radio-frequency identification (RFID) applications: a brief introduction. Adv. Eng. Inf. **21**(4), 350–355 (2007)
2. Wankhede, K., Wukkadada, B., Nadar, V.: Just walk-out technology and its challenges: a case of Amazon go. In: 2018 International Conference on Inventive Research in Computing Applications (ICIRCA). IEEE (2018)
3. Wu, B.-F., et al.: An intelligent self-checkout system for smart retail. In: 2016 International Conference on System Science and Engineering (ICSSE). IEEE (2016)
4. Redmon, J., Farhadi, A.: YOLO9000: Better, faster, stronger. arxiv (2016). arxiv preprint arXiv:1612.08242
5. Redmon, J., et al.: You only look once: unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
6. Karlinsky, L., et al.: Fine-grained recognition of thousands of object categories with single-example training. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
7. George, M., et al.: Fine-grained product class recognition for assisted shopping. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (2015)
8. Tonioni, A., Di Stefano, L.: Product recognition in store shelves as a sub-graph isomorphism problem. In: Battiato, S., Gallo, G., Schettini, R., Stanco, F. (eds.) ICIAP 2017. LNCS, vol. 10484, pp. 682–693. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-68560-1_61
9. Franco, A., Maltoni, D., Papi, S.: Grocery product detection and recognition. Expert Syst. Appl. **81**, 163–176 (2017)

10. Solti, A., et al.: Misplaced product detection using sensor data without planograms. Decision Support Syst. **112**, 76–87 (2018). S0167923618301039
11. Papadopoulos, D.P., et al.: We don't need no bounding-boxes: training object class detectors using only human verification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
12. Mettes, P., van Gemert, J.C., Snoek, C.G.M.: Spot on: action localization from pointly-supervised proposals. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9909, pp. 437–453. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46454-1_27
13. Papadopoulos, D.P., et al.: Training object class detectors with click supervision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
14. Papadopoulos, D.P., Clarke, A.D.F., Keller, F., Ferrari, V.: Training object class detectors from eye tracking data. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 361–376. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_24
15. Vedaldi, A., Bilen, H.: Weakly supervised deep detection networks. In: Institute of Electrical and Electronics Engineers (2016)
16. Kantorov, V., Oquab, M., Cho, M., Laptev, I.: ContextLocNet: context-aware deep network models for weakly supervised localization. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9909, pp. 350–365. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46454-1_22
17. Zhu, Y., et al.: Soft proposal networks for weakly supervised object localization. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
18. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
19. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
20. Van Horn, G., et al.: The inaturalist species classification and detection dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
21. Russakovsky, O., et al.: Imagenet large scale visual recognition challenge. Int. J. Comput. Vis. **115**(3), 211–252 (2015)
22. King, D.E.: Dlib-ml: a machine learning toolkit. J. Mach. Learn. Res. **10**(7), 1755–1758 (2009)
23. Howard, A.G., et al.: MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
24. Chollet, F.: Xception: deep learning with depthwise separable convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
25. Liu, S., Huang, D., Wang, Y.: Receptive field block net for accurate and fast object detection. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11215, pp. 404–419. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01252-6_24
26. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2015)
27. Wei, X.-S., et al.: RPC: A Large-Scale Retail Product Checkout Dataset. arXiv preprint arXiv:1901.07249 (2019)