Innovate the Retail Industry with Automatic Checkout System

Qian Zian 20412089 | Ren Xuanchi 20493435 | Wang Zhiwei 20412807

1. Introduction

Checkout is always time-consuming for customers in the supermarkets. Through history, technologies such as the barcode, QR code were developed to improve the efficiency of checkout. In recent years with the rapid development of computer vision technique, we see an opportunity to improve the checkout experience even further by fully automating the process. Customers can check out just by using their phone to take a photo of their shopping items without waiting in the long check out lines. We believe this is the future of checkout technology for the retail industry. In this project, we will use computer vision technique to implement an automatic checkout (ACO) system.

To gain necessary background knowledge, we spent three weeks reading papers together with regard to Faster R-CNN [8], FPN [5] and Mask R-CNN [3], which are some of the state-of-art technique in object detection. After this, we returned to our problem by first understanding the dataset we use and writing the API for data retrieval. During the last two weeks, we managed to use Faster R-CNN and FPN to train a full checkout model with reasonable results, as well as use data augmentation technique to synthesize checkout images for better training. The intermediate results of our work are presented in the later section of this report. For the next stage, we plan to use the synthesized images and the segmentation mask generated by ourselves for training. Further optimization on the model architecture would be done by adding GIoU [9] and soft NMS [1] to the mask R-CNN framework.

2. Problem Statement

As shown in Fig.1, the model should be able to output all the item categories in the image, the exact count of each category and the total price. Checkout accuracy (cACC) [10] is introduced as the main evaluation metric we use for the ACO task. Given i images and k different product category, ($CD_{i,k}$) represents the the counting error for a specific product category in an image, where Pi,k is the predicted label and $GT_{i,k}$ is the ground truth label.

$$CD_{i,k} = |P_{i,k} - GT_{i,k}|,$$

Then, CD_i is defined to represent the total counting error

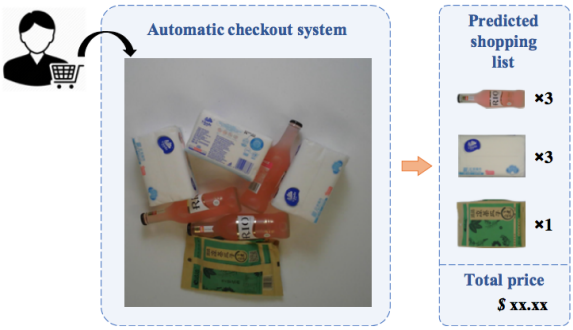

Figure 1. Illustration of the expected output.

for the i-th image. $CD_i = 0$ suggests a fully correct prediction on the image.

$$CD_{i,k} = \sum_{k=1}^{K} CD_{i,k},$$

Checkout accuracy is then defined to evaluate the prediction accuracy of the model. $\delta(\cdot)$ returns 1 if and only if $\sum_{k=1}^{K} CD_{i,k} = 0$; This means that a prediction list is considered successful only if its exactly the same with the ground truth shopping list.

$$cAcc = \frac{\sum_{i=1}^{N} \delta(\sum_{k=1}^{K} CD_{i,k}, 0)}{N},$$

The most critical challenge of ACO is that the system should be able to reach approximately 100% cAcc for real-world production. Miss classified item or wrong count of a product category would bring serious problems to retailers as well as customers. Besides, the large scale of classes (200 classes in our dataset), fine-grained nature of the product categories and continuous update of new categories also bring difficulty on implementing ACO. These challenges are not well studied in the current research landscape, which requires us to use a trial-and-error approach on different techniques.

Figure 2. Original Image Using Salient Object Detection, and finally matting the object.

3. Technical Approach

In this section, we will explain in details about the technical approach we plan to use to solve ACO problem. Some of the approaches have already been implemented with results shown in the next section.

3.1. Data Augmentation

The ACO dataset [10] we use contains 60000 training images. However, each training image only contains a view of a single product. This have brought problems for the training. Firstly, as we are using Faster R-CNN and FPN in this phase, using this kind of training data will cause training failure of RPN in Faster R-CNN because all those objects are in the center of the image. The network may only be trained to detect objects in the center. Secondly, using images with single object to train our network may not help much on predicting multiple objects in a image. Thirdly, the background of those training images are different from the realistic checkout environment, which may decrease prediction accuracy when applying our model to reality. Fourthly, there is no segmentation information of all the data. Thus, the goal of our data augmentation is to synthesize considerable amount of multi-object training images which assemble real-world checkout environment. The technical approaches we use for data augmentation are introduced below.

1. Matting using bounding box of the training data. (Already completed)

   The naive way we have considered is matting the object using the bounding box provided, then copy and paste the bounding box image to a white background. However, the problem here is that the image we synthesize seems not realistic, as shown in Figure 3. and all the object are squared, which may lead our network to memorize this pattern during training.

2. KNN Matting (Already completed)

   The second approach is using matting algorithms. We tried KNN Matting [2], but there are two problems on this algorithm. The first one is that this matting algorithm requires trimaps, which may take a long time for us to label them on our own data. The second problem is that as we run KNN matting on our data, we find that the runtime of KNN matting is very long. Each image needs 1-2 minutes process time, which can take us 60 days to mat all of the 60000 images.

Figure 3. Our Matting Using Bounding Box, which is not that realistic.

3. Salient Object Detection (Already completed)

   The third approach we tried is to use salient object detection for matting. Unlike the traditional salient object detection algorithm, we try to use a deep learning based salient object detection proposed in [4]. As we mentioned before, our dataset does not contain segmentation information, which means we don't have ground truth label to train this network. Thus, we used a pre-trained model instead to generate the salient map on our own training images. By feeding bounding box images to the model, it seems that the model cannot clearly distinguish foreground and background. For example, product logos are always matted out. To solve this problem, we first double the size of the bounding box to provide more information of the background, and then grey scaled the part outside the central bounding box in order to increase contrast. After generating the salient map of the images in this way, we re-sized it into 3 channels and multiplied it by the original image to get the object we want. More results

will be shown in the next part in detail.

4. Segmentation generation based on salient map (Further Plan)

   After we get the salient map of the single object images, it is possible for us to generate the segmentation information using image processing algorithms for the use of Mask-RCNN.

5. Cycle-GAN [11] for generating realistic training images(Further Plan)

   From our intermediate results shown in Figure 7. we can see that the images we generate still seem not quite realistic, which means there is a gap between our training data and real world test data. Thus, we plan to use Cycle-GAN with real world checkout images and our synthesized images to make our synthesized images closer to the real world checkout images.

3.2. Network Structure and Algorithms

1. Choice of main framework

   There are many kinds of mainstream frameworks for object detection, and these network architectures can be classified as one-stage network and two-stage network. It is unwise to compare results side-by-side from different papers since experiments are done in different settings. Nevertheless, there's a general conclusion that one-stage networks like SSD [6] and YOLO [7] run faster and two-stage networks like Faster R-CNN [8] have higher prediction accuracy. As mentioned in the problem statement, ACO task requires high prediction accuracy, which makes Faster R-CNN a more favorable framework for this task.

2. Why FPN and ROI-align?

   Due to the absence of segmentation information of our dataset, we use Faster R-CNN [8] rather than Mask R-CNNhe2017mask for now. To improve the prediction accuracy of the model, FPN [5] and ROI-align are added to the Faster R-CNN framework.

   Feature Pyramid Network(FPN) [5] combines low-resolution, semantically strong features with high-resolution, semantically weak features via a top-down pathway and lateral connections. This feature pyramid has rich semantics at all levels and is built quickly from a single input image scale, which contributes to higher prediction accuracy. For ROI-align, in each ROI bin, the value of the four regularly sampled locations are computed directly through bi-linear interpolation. This help avoid the misaligned problem which occurs in ROI-pooling.

AP	IoU	Area	maxDets
0.858	0.50:0.95	all	100
0.994	0.50	all	100
0.968	0.75	all	100
0.000	0.50:0.95	small	100
0.000	0.50:0.95	medium	100
0.858	0.50:0.95	large	100

Table 1. Average Precision of Our Result.

AR	IoU	Area	maxDets
0.469	0.50:0.95	all	1
0.897	0.50:0.95	all	10
0.897	0.50:0.95	all	100
0.000	0.50:0.95	small	100
0.000	0.50:0.95	medium	100
0.897	0.50:0.95	large	100

Table 2. Average Recall of Our Result.

3. Future Prospect

   As mentioned previously, ACO problem requires high prediction accuracy. Though our network makes really small amount of error (Table 1 and Table 2), we will further optimize it with soft NMS [1] and GIoU [9]. Soft NMS [1] decreases the detection scores as an increasing function of IoUs between the target proposals and the overlapped ones, rather than setting scores of the overlapped proposals to zero. GIoU [9] overcomes the disadvantages of IoU while keeping the useful characteristics of IoU.

4. Intermedia Result

4.1. Data Augmentation

Up to now, we have already generated the salient map of all 60000 training images with some of them shown in figure.6. We use those salient maps to generate a large amount of synthesized images with labels as shown in figure.7. However, due to the limitation of time, we didn't use them for training.

4.2. Network Structure and Algorithms

The results of our network up to now are shown in Table.1 and Table.2, where AR is average recall, AP is average precision and area is the size of the bounding box we want to detect.'small' means the size of an object is less than 322. 'medium' means the size of an object is less than 962 and greater than 322 and 'large' means the size of an object is greater than 962. There's no object with size less than 322, so the precision and recall are both 0. MaxDets means the threshold of max number of detection.

Figure 4. Visualized Images of our Test Result.

Figure 5. Failure Case.Two glues on the left are classified as one.

Since the synthesized images are just ready for use, the current model takes test data for training and validation data for testing. This is a temporary setting for us to set up model architecture. We'll use synthesized images for training in the next step. Some of the visualized test images of our current model are shown in figure 4, which produce the exact right prediction. However, there are still some failure examples, one of which is shown in figure5. The failure is largely caused by dense placement and fine-grained differences. These will be the focus for future improvement.

5. Conclusion

Till now, we have made progress on understanding the background knowledge of object detection, writing API for data retrieval, synthesizing training images and training a full ACO system with reasonable prediction results. These work provide a solid base for the further optimization of the model. For the next stage, there will be two focus of our work: (1) improve the training data by adding synthesized images and segmentation information; (2) optimize network architecture by using GIoU and soft nms. We aim to achieve at least 70% checkout accuracy at the end of this project. (Github repository: https://github.com/zqianaa/comp4901j)

Figure 6. The Image on the Left is the Original Single Object Images, The Image on the Right is the Salient Map we Generate

Figure 7. Our Synthesis Images Using Single Object Training Images

References

[1] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5561–5569, 2017.

[2] Q. Chen, D. Li, and C.-K. Tang. Knn matting. *IEEE transactions on pattern analysis and machine intelligence*, 35(9):2175–2188, 2013.

[3] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[4] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, and P. H. Torr. Deeply supervised salient object detection with short connections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3203–3212, 2017.

[5] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2117–2125, 2017.

[6] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[7] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[8] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[9] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese. Generalized intersection over union. June 2019.

[10] X.-S. Wei, Q. Cui, L. Yang, P. Wang, and L. Liu. Rpc: A large-scale retail product checkout dataset. *arXiv preprint arXiv:1901.07249*, 2019.

[11] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.