Unitail: Detecting, Reading, and Matching in Retail Scene Supplementary Materials

Fangyi Chen[1], Han Zhang[1], Zaiwang Li[2], Jiachen Dou[1], Shentong Mo[1], Hao Chen[1], Yongxin Zhang[3], Uzair Ahmed[1], Chenchen Zhu[1], and Marios Savvides[1]

[1] Carnegie Mellon University, Pittsburgh PA 15213, USA
[2] University of Pittsburgh, Pittsburgh PA 15213, USA
[3] Tsinghua University, Beijing 100084, China
{fangyic,hanz3,jiachend,shentonm,haoc3,uzaira,marioss}@andrew.cmu.edu
zal17@pitt.edu yx-zhang20@mails.tsinghua.edu.cn chenchez@alumni.cmu.edu

A Datasets Comparison

Dataset	#Image	#Instance	Box Type	#Category	Det	Rec	Text
Grozi-3.2k	9,030	11,585	AABB	80	✓		
Grocery Shelves	354	13,000	AABB	10	✓		
SKU110k	11,748	1,730,996	AABB	1	✓		
SKU110k-r	11,748	1,731,762	RBOX	1	✓		
Locount	50,394	1,905,317	AABB	140	✓		
RPC	30,000	367,935	AABB	200	✓	✓	
Grozi-120	11,870	-	-	120		✓	
SOIL-47	987	-	-	47		✓	
SuperMarket	2,633	-	-	15		✓	
Freiburg	5,021	-	-	25		✓	
Product10K	150,000	-	-	10k		✓	
Unitail(ours)	12,244	1,777,108	QUAD	1454	✓	✓	✓

Table 1: Comparison of related benchmarks.

B Annotation

B.1 Unitail-Det

Annotation Method We consider quadrilaterals as proper fits to products. The bounding box is common for localizing objects and reflects their shape in detection tasks. Axis-aligned rectangles are popular because they satisfy the minimum requirement for learning targets with minimum labeling efforts. Annotations with more accurate localization and appearance alignment are needed in this task. While segmentation masks are another level of accuracy for annotating

Fig. 1: Annotation examples of the Unitail-Det. Top-left: small products are ignored by masks (red bounded regions). Top-right: frontal faces represent products. Bottom: quadrilaterals on irregular-shaped products.

scene objects, they are not cost-friendly, and the direct regression on bounding boxes is easier for densely-packed products than the segmentation methods verified in the benchmark.

To ensure the quality of annotation, annotators follow a strictly defined guide. Illustrated in Fig.1, products localized far away from the camera and with a size less than 8×8 pixels are not treated as positive. Instead, we annotate ignoring masks covering the distant regions for these products. Cuboid and cylinder (boxes, cans, bottles), as the majority in stores, enjoy normal quadrilaterals defined in the paper; spherical and cones whose corners are difficult to identify, along with irregularly shaped products and distorted bags are expected to be affine-transformed back into an upright position. In those cases, we first draw the minimum AABB to cover them and then adjust the four corners according to the camera perspective. Only the visible part is annotated if another product or tag blocks one product. Many products have multiple faces observable, and only the frontal face of the products is practically annotated. Labelme [15] is applied as an annotation tool.

Although many existing methods [9, 11, 20] re-order the regression targets (four corner points) to favour the loss convergence regardless of the original

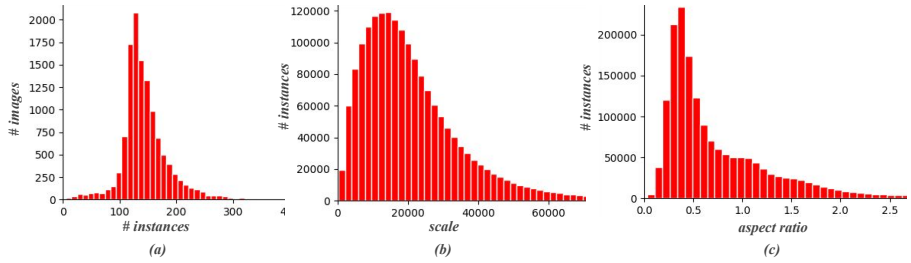

Fig. 2: Histograms. (a) Instance density. (b) Instance scales. (c) Instance aspect ratio

order in ground-truths, we define the first corner point $(x1, y1)$ as the top-left corner.

Statistics Fig.2 (a) illustrates the number of QUADs in each image. The mean and standard deviation is 145 and 46, respectively. An image in QuadRetail may contain at least 5 QUADs and up to 744 QUADs. Despite the density, the overlap among QUADs is not severe due to the annotation standard.

Fig.2 (b) illustrates the scales of QUADs. The average scale is 22393.7 (149.6^2) over the QuadRetail. The minimum and maximum are 17^2 and 1938^2, respectively. The average image width is 2466.8, and the average height is 3288.1.

The aspect ratio of a rectangle is commonly defined as $\frac{w}{h}$. We measure the aspect ratio of QUAD as in Eq.1

$$ratio = \sqrt{\frac{t \cdot b}{l \cdot r}} \tag{1}$$

where t, b, l, r are lengths of top, bottom, left, and right edge, respectively. Fig.2 (c) illustrates the aspect ratio. Most ratios are around 0.3 - 0.6, which is in line with the practical observation. There are also QUADs with extreme AR ($<$0.05) and ($>$38).

The interior angles of any convex QUAD add up to 360 degrees of arc. The standard deviation of these angles (std_a) is a qualified reflection of the QUAD shape. For rectangles, $std_a = 0$. For a extreme QUAD looks close to a line segment, $std_a = 90$. In the origin-domain of QuadRetail, the $std_a = 6.24$. In the cross-domain, the $std_a = 12.73$. This means the images from the cross-domain are from taken from tougher angles.

B.2 Unitail-OCR

Unitail-OCR consists of a gallery and a testing suite to support text detection, text recognition, and product recognition.

Annotation Method We start by cropping products from the Unitail-Det cross-domain. Since the crops are in quadrilaterals, they are further transformed to form upright appearances. Products with scales larger than 15×15 are further selected. The categorization is organized by two hierarchical stages. In the first stage, ten supercategories (food, dairy, paper goods, canned, produce, clothing, Technology, Pharmacy, Care, other) are defined to classify the products coarsely. In the second stage, we first apply a strong pre-trained model trained with a bag of tricks to group the product into 6k clusters and correct them by human annotators. During the correction, we do not accept blurred products that are hard to identify. Many fine-grained categories rely on textual information, and the blurred ones that are difficult to read are filtered out. After the categorization, we label each product with word-level text boxes, following the annotation method in ICDAR2015 [4]. We annotate 29681 text regions from 4466 products as quadrilateral text boxes. The bounding boxes are classified as legible or illegible. The alphanumeric transcriptions $0 \sim 9, a \sim z$ are annotated for the legible ones. A vocabulary covers all words in the gallery and the testing suite are attached with the dataset.

C Baseline Implementation

C.1 Off-the-shelf Algorithms

We conduct experiments using multiple codebases including mmdetection [1] for FCOS, SAPD, ATSS; mmocr [5] for DBNet, FCENet, PSENet, CRNN, NRTR, RobustScanner, SAR, SATRN, ABINet; AlphaRotate [17] for RIDet and RSDet; maskrcnn benchmark[8] for maskrcnn and Gliding Vertex; timm [16] for efficientnet, ResNet, and ResNet-IBN. To be more specific, models for product detection tasks are trained on 4 NVIDIA Titan RTX GPUs with two images per GPU. For fair comparison, the training schedule is 12 epochs with an initial learning rate of 0.01 divided by 10 at the 9th and the 11th epoch. Unless otherwise specified, the input images are scaled to 1200 and randomly horizontally flipped without any other augmentation. The Convex Hull and Shoelace Formula are implemented in CUDA to calculate the exact IOU of QUADs. Up to 400 detections per image are allowed to evaluate. For text detection models, we respect their optimized training setting for ICDAR2015 based on their officially released paper or code, but change the input image size to 1333×800. For text recognition, we finetune the publicly available weights on Unitial-OCR for 10 epochs.

C.2 RetailDet and RetailDet++

Base Network Our design of base network applies prior-art DenseBox-style head [3] to multiple feature pyramid levels. The feature pyramid is generated via feature pyramid network (FPN) [6] which utilizes a deep convolutional network as the backbone. As an image is fed into the backbone, several feature maps are extracted to compose the initial feature pyramid. This work adopts the ResNet

family as the backbone, and the extracted feature maps are from C_3 to C_5. The feature maps after FPN are denoted as P_3, P_4, P_5. An anchor-free detection head is attached then. The head contains two branches. One is a binary classification branch to predict a heatmap for product/background. Another is a regression branch to predict the offset from the pixel location to the four corner points of the QUAD. Each branch consists of 3 stacks of convolutional layers followed by another c channel convolutional layer, where c equals to 1 for the classification branch and 8 for the regression branch.

Corner Refinement Module RetailDet++ is the RetailDet enhanced with Corner Refinement Module (CRM) and deeper backbone. Here, we introduce the CRM. For each predicted QUAD from the RetailDet, we get the locations of its four corners and center. Then we apply the bilinear interpolation to extract feature of 5 points (4 corners, one center) from the feature map generated by the 3rd stacked convolution in the regression branch. These features are concatenated and fed into a 1×1 convolutional layer to predict the offsets between ground-truth and the former predictions. The same operation and convolution are also inserted in the classification branch to predict retail/background as a 2nd-stage classification. During testing, we combine the regression results from the two stages but only use the classification result from the first stage. CRM shares the spirits with Faster-RCNN[14], BorderDet[12] and Reppoints [18], but we find that the 5 points as mentioned above are enough for quadrilateral products, and the 2nd-stage classification helps training though not involved in testing.

Losses During training, we first shrink QUADs by a ratio $\alpha = 0.3$ according to the gravity centers. If one feature pixel locates inside the shrunk QUAD, the pixel is considered responsible for learning the ground-truth. We utilize $focal\ loss$ [7] for classification and $SmoothL_1\ loss$ for regression, and we reweight both losses by the production of quad-centerness and level reweighting factor F. The total loss is the summation of the classification and regression losses. If two-stage, additional $focal\ loss$ and $L_1\ loss$ for CRM are added to the total loss.

D Discussion

D.1 Proof: Eq.2 Is Equivalent to Eq.2 on Rectangles

When QUAD is specialized to Rectangles, in Eq.2, $d_g^l = d_g^r$, $d_p^l + d_p^r = 2d_g^l$, so if $d_p^l < d_g^l$, then $d_p^r > d_g^l = d_g^r$; if $d_p^l > d_g^l$, then $d_p^r < d_g^l = d_g^r$. Thus, $\frac{\min(d_p^l, d_g^l)}{\max(d_p^l, d_g^l)} \cdot \frac{\min(d_p^r, d_g^r)}{\max(d_p^r, d_g^r)} = \frac{\min(d_p^l, d_p^r)}{d_g^l} \cdot \frac{d_g^r}{\max(d_p^l, d_p^r)} = \frac{\min(d_p^l, d_p^r)}{\max(d_p^l, d_p^r)}$, similarly to d^t and d^b, then, Eq.2 is mathematically equivalent to Eq.1.

D.2 Analysis on Soft Selection

Soft Selection is a loss-based strategy, where training losses of ground-truths indicate their pyramid level. It first assigns each object to all pyramid levels

P_3, P_4, P_5 and calculates $loss_l$ for each level P_l. l=3,4,5. Then, the level that produces the minimal loss is converted to a one-hot vector, i.e., (1,0,0) if the minimal loss is from P_3; and (0,1,0) if it is from P_4, and so on. The vector is used as the ground-truth to train an auxiliary network that simultaneously predicts a vector (F_3, F_4, F_5). Each element F_l is a down-weighting factor for $loss_l$. The final loss of each object is $\sum_l (F_l \cdot loss_l)$.

By Soft Selection, the minimal loss from level l indicates that the auxiliary network is trained to generate a relatively larger F_l, but we find the loss not independent of scales. On the contrary, object scale inherently determines which level will produce the minimal loss. We claim the reason as follows. First, when assigning objects (e.g. object A with size 8×8 and B with size 16×16) to pyramid, their regression targets (denoted as T_A, T_B) are normalized by the level stride. Specifically, on a lower level like P_3, the target is divided by stride 8, while on a higher level like P_4, the target is divided by 16, and so on. Therefore, when assigning A to P_3 and P_4, T_A is 1×1 and 0.5×0.5, respectively; when assigning B, T_B is 2×2 and 1×1, respectively. Note that all levels share the detection head. Apparently, the combination of $T_A = 1 \times 1$ and $T_B = 1 \times 1$ leads to the smallest regression difficulty for the regression head. Naturally, it produces minimal regression losses, which means the smaller object is assigned to a lower level. Second, since A has a smaller scale, it requires more local fine-grained information beneficial for classification, which is more available from high-resolution lower levels. In comparison, B has a larger scale and needs a larger receptive field, which is more available from higher levels. Therefore, the "loss-based" Soft Selection, in essence, follows the scale-based law.

Nevertheless, why does Soft Selection outperforms scale-based strategies? We credit the improvement to its loss-reweighting mechanism. This mechanism involves multiple levels during training and reweights the loss in terms of the regression and classification difficulties, making optimization easier. Since the pyramid is discrete, if an object scale falls into the gap of two adjacent levels, both levels' difficulties will be similar. The auxiliary network has opportunities to learn to predict proper F_l for both levels. The analysis motivates us to abandon the auxiliary network and design Soft Scale (SS).

E Additional Results

E.1 RetailDet

On SKU110k-R The result of RetailDet on SKU110k-R is compared with other methods in Table.2. RetailDet outperforms the state-of-the-art detectors CenterNet[19], DRN [10] and CFA [2] on SKU110k-r where products are in RBOX style. Following CFA, we use multi-scale training.

Ablation Study Table.3 shows the improvement of each component brings to RetailDet. The Quad-Centerness (QC), Soft-Scale (SS) and Corner Refinement Module (CRM) gradually improve the mAP by 2.1, 1.0, 2.4 in the origin domain

Method	mAP	AP75	AR300
YoloV3-Rotate[13]	49.1	51.1	58.2
CenterNet-4point[19]	34.3	19.6	42.2
CenterNet[19]	54.7	61.1	62.2
DRN[10]	55.9	63.1	63.3
CFA[2]	57.0	63.5	63.9
RetailDet	**57.5**	**65.5**	**64.3**

Table 2: Detection performance on SKU110k-r.

Base	QC	SS	CRM	mAP	AP50	AP75
✓				58.3	87.0	69.5
✓	✓			60.4	89.3	71.5
✓	✓	✓		61.4	90.3	72.6
✓	✓	✓	✓	63.8	91.2	75.8

Table 3: Ablation study on the Unitail-Det val set. QC: Quad-Centerness. SS: Soft Scale. CRM: Corner Refinement Module.

and 1.8, 0.6, 3.2 in the cross domain. And the improvement is consistent under different IOU thresholds.

E.2 Inference Speed

We report the inference speed of the key models in Table 4

Methods	Backbone	Task	FPS
RetailDet	ResNet101	Product Detection	6.3
DBNet	ResNet50	Text Detection	27.6
ABINet	ResNet-ABI	Text Recognition	41.4
Visual+Text*	-	Product Matching	65.1

Table 4: Speed tested on single 2080Ti. * pipeline is accelarated without losing accuracy by applying only text models on visually low-confidence products.

E.3 Qualitative Results

We visualize the product detection results in Fig.3, Fig.4, and Fig.5. The detector is our RetailDet two-stage variant with ResNext101. The average testing speed is 2.8FPS. We only visualize the QUADs with confident scores higher than 0.3.

We show some failure cases in the cross domain in Fig.6. (a)(b) Unseen categories. (c)(d) Tough camera angles.

We show the OCR results from DBNet and ABINet in Fig.7.

References

1. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., et al.: Mmdetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019)
2. Guo, Z., Liu, C., Zhang, X., Jiao, J., Ji, X., Ye, Q.: Beyond bounding-box: Convex-hull feature adaptation for oriented and densely packed object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 8792–8801 (June 2021)
3. Huang, L., Yang, Y., Deng, Y., Yu, Y.: Densebox: Unifying landmark localization with end to end object detection. arXiv preprint arXiv:1509.04874 (2015)
4. Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., Ghosh, S., Bagdanov, A., Iwamura, M., Matas, J., Neumann, L., Chandrasekhar, V.R., Lu, S., Shafait, F., Uchida, S., Valveny, E.: Icdar 2015 competition on robust reading. In: Proceedings of the 2015 13th International Conference on Document Analysis and Recognition (ICDAR). p. 1156–1160. ICDAR '15, IEEE Computer Society, USA (2015). https://doi.org/10.1109/ICDAR.2015.7333942, https://doi.org/10.1109/ICDAR.2015.7333942
5. Kuang, Z., Sun, H., Li, Z., Yue, X., Lin, T.H., Chen, J., Wei, H., Zhu, Y., Gao, T., Zhang, W., Chen, K., Zhang, W., Lin, D.: Mmocr: A comprehensive toolbox for text detection, recognition and understanding. arXiv preprint arXiv:2108.06543 (2021)
6. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR (2017)
7. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
8. Massa, F., Girshick, R.: maskrcnn-benchmark: Fast, modular reference implementation of Instance Segmentation and Object Detection algorithms in PyTorch. https://github.com/facebookresearch/maskrcnn-benchmark (2018), accessed: [Insert date here]
9. Ming, Q., Miao, L., Zhou, Z., Yang, X., Dong, Y.: Optimization for arbitrary-oriented object detection via representation invariance loss. IEEE Geoscience and Remote Sensing Letters pp. 1–5 (2021). https://doi.org/10.1109/LGRS.2021.3115110
10. Pan, X., Ren, Y., Sheng, K., Dong, W., Yuan, H., Guo, X., Ma, C., Xu, C.: Dynamic refinement network for oriented and densely packed object detection pp. 1–8 (2020)
11. Qian, W., Yang, X., Peng, S., Yan, J., Guo, Y.: Learning modulated loss for rotated object detection. Proceedings of the AAAI Conference on Artificial Intelligence **35**(3), 2458–2466 (May 2021), https://ojs.aaai.org/index.php/AAAI/article/view/16347
12. Qiu, H., Ma, Y., Li, Z., Liu, S., Sun, J.: Borderdet: Border feature for dense object detection. CoRR **abs/2007.11056** (2020), https://arxiv.org/abs/2007.11056
13. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
14. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Advances in neural information processing systems. pp. 91–99 (2015)
15. Wada, K.: labelme: Image polygonal annotation with python. https://github.com/wkentaro/labelme (2018)

16. Wightman, R.: Pytorch image models. https://github.com/rwightman/pytorch-image-models (2019). https://doi.org/10.5281/zenodo.4414861
17. Yang, X., Zhou, Y., Yan, J.: Alpharotate: A rotation detection benchmark using tensorflow. arXiv preprint arXiv:2111.06677 (2021)
18. Yang, Z., Liu, S., Hu, H., Wang, L., Lin, S.: Reppoints: Point set representation for object detection. arXiv preprint arXiv:1904.11490 (2019)
19. Zhou, X., Wang, D., Krähenbühl, P.: Objects as points. CoRR **abs/1904.07850** (2019), http://arxiv.org/abs/1904.07850
20. Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., Liang, J.: EAST: an efficient and accurate scene text detector. CoRR **abs/1704.03155** (2017), http://arxiv.org/abs/1704.03155

Fig. 3: Visualization of high difficulty detection result from the RetailDet.

Fig. 4: Visualization of medium difficulty detection result from the RetailDet.

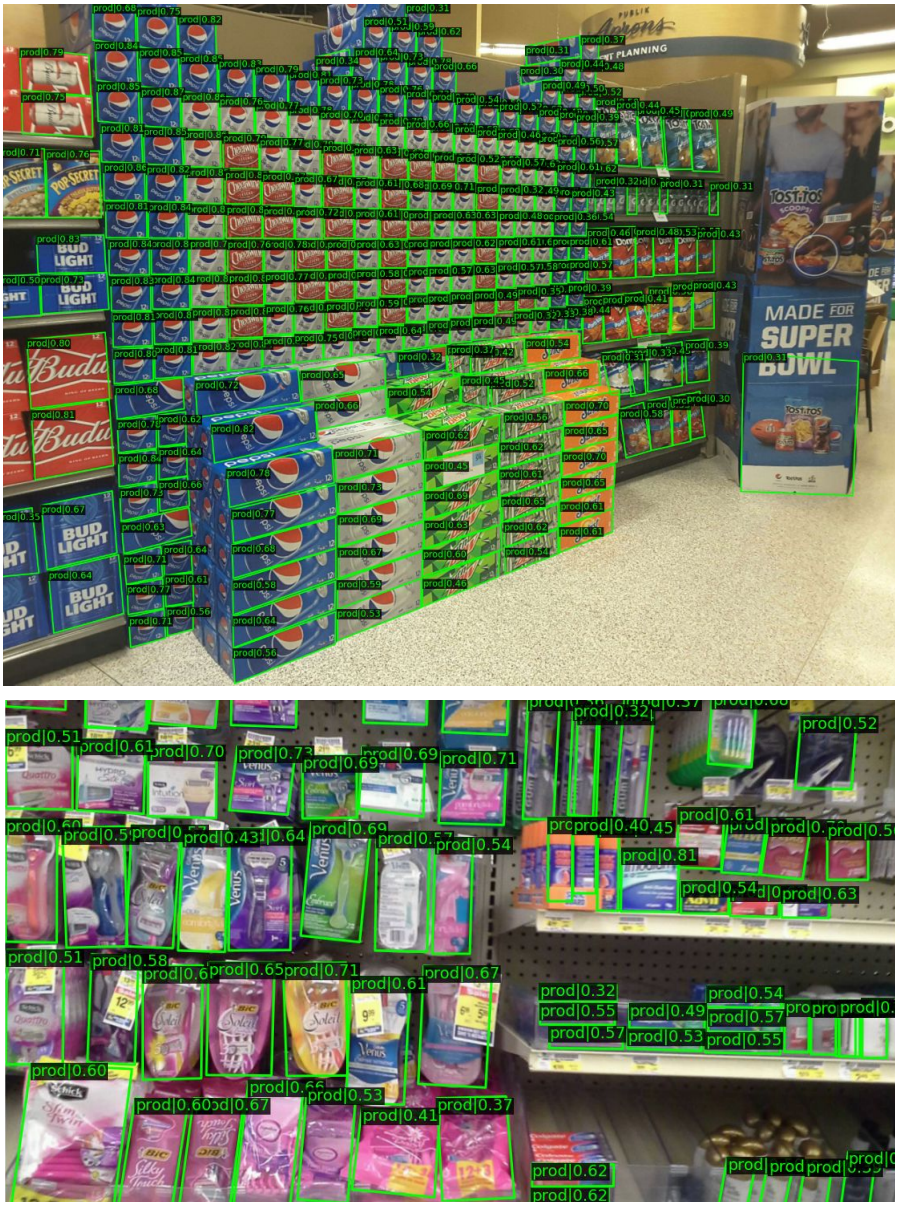

Fig. 5: Visualization of low difficulty detection result from the RetailDet.

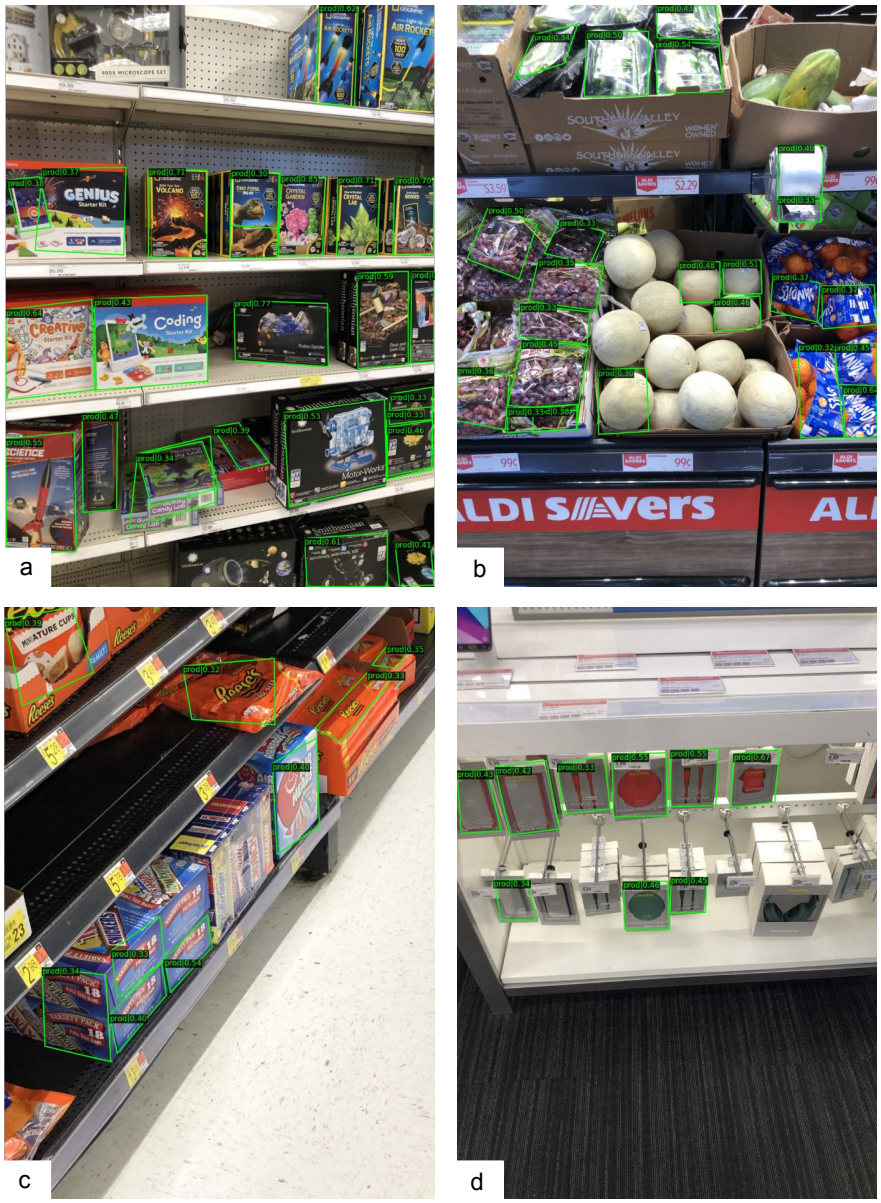

Fig. 6: Visualization of failure cases in cross domain. (a)(b) Unseen categories. (c)(d) Tough shooting angles.

Fig. 7: Visualization of OCR results. Upper ones of each section are from models trained on ICDAR2015, and lower ones are on Unitail-OCR.