Fast, Accurate Barcode Detection in Ultra High-Resolution Images

Jerome Quenum, Kehan Wang, and Avideh Zakhor

Department of Electrical Engineering and Computer Science

University of California, Berkeley

{jquenum, wang.kehan, avz}@berkeley.edu

1. OVERVIEW

Object detection in Ultra High-Resolution (UHR) images has long been a challenging problem in computer vision due to the varying scales of the targeted objects. When it comes to barcode detection, resizing UHR input images to smaller sizes often leads to the loss of pertinent information, while processing them directly is highly inefficient and computationally expensive. In this work, we propose using semantic segmentation to achieve a fast and accurate detection of barcodes of various scales in UHR images. Our pipeline involves a modified Region Proposal Network (RPN) on images of size greater than 10k×10k and a newly proposed Y-Net segmentation network, followed by a post-processing workflow for fitting a bounding box around each segmented barcode mask. The end-to-end system has a latency of 16 milliseconds, which is 2.5× faster than YOLOv4 and 5.9× faster than Mask R-CNN. In terms of accuracy, our method outperforms YOLOv4 and Mask R-CNN by a mAP of 5.5% and 47.1% respectively, on a synthetic dataset. We have made available the generated synthetic barcode dataset and its code at http://www.github.com/viplabB/SBD/ and the 2021 IEEE *International Conference on Image Processing* (ICIP) publication on this work could be found at https://ieeexplore.ieee.org/document/9506134/.

2. APPROACH

Our proposed method consists of three stages: the modified Region Proposal Network stage, our Y-Net * segmentation network stage, and the bounding box extraction stage.

2.1 Modified Region Proposal Network

Region proposals have been influential in computer vision and more so when it comes to object detection in UHR images. It is common in UHR images that barcodes are clustered in a small region of the image. To filter out most of the non-barcode backgrounds, we modified the RPN introduced in Faster R-CNN[2] to propose regions of barcodes for our next stages. By first transforming the UHR input image to an LR input image of size 256×256, the RPN was trained to identify blobs in LR images. Once a bounding box is placed around the identified blobs, the resulting proposed bounding box is remapped to the input UHR image by a perspective transformation, and the resulting regions are cropped out.

2.2 Y-Net Segmentation Network

Y-Net is made out of 3 main modules distributed in 2 branches: a Regular Convolutional Module which constitutes the left branch, and a Pyramid Pooling Module, along with a Dilated Convolution Module which after concatenation and convolution constitute the right branch.

The ***Regular Convolution Module*** allows the model to learn general pixel-wise information anywhere in the input image and consists of convolutional and pooling layers.

The ***Dilated Convolution Module*** takes advantage of the fact that barcodes have alternating black and white rectangles to learn sparse features in their structure.

*Our Y-Net architecture resembles the English alphabet letter "Y" and differs from[1] which used a pre-trained encoder network that is augmented with an untrained mirrored network and a decoder network.

The ***Pyramid Pooling Module*** allows the model to learn global information about potential locations of the barcodes at different scales and its layers are concatenated with the layers on the dilated convolution module in order to preserve the features extracted from both modules.

The resulting feature maps from the right branch are then added to the output of the Regular Convolution Module, which allows for the correction of features that would have been missed by either branch. In other words, the output of each branch constitutes a residual correction for the other thereby refining the result at each node. The nodes are then up-sampled and concatenated with transposed convolution feature maps of the corresponding dimension.

2.3 Bounding Box Extraction

Since some images contain barcodes that are really close to each other, their Y-Net outputs reflect the same configuration which makes the extraction of individual barcode bounding boxes complex. To separate them effectively, we perform an erosion, contour extraction, and bounding box expansion with a pixel correction margin. This post-processing stage of our pipeline has an average processing time of 1.5 milliseconds (ms) because it is made of a set of Python matrix operations to efficiently extract bounding boxes from predicted masks.

3. RESULTS

We use Code 39, Code 93, Code 128, UPC, EAN, PD417, ITF, Data Matrix, AZTEC, and QR among others. We model the number of barcodes in a given image using a Poisson process and a combination of perspective transforms is used to make the barcodes vary in shape and position from one image to the other. We have also added random black blobs at random locations on the original UHR and LR canvases. The real UHR barcodes dataset obtained from Amazon.com, Inc is made of 3.8 million UHR images of resolution up to $30k \times 30k$ grayscale images and could not be released due to confidentiality reasons. Additionally, the Muenster and Artelab datasets are used with some data augmentation schemes for more samples.

For the RPN, we accumulated the number of bounding boxes inside the proposed regions and divided it by the total number of ground truth bounding boxes. Our implementation yields an accuracy of 98.03% on the synthetic dataset at 10 ms per image and 96.8% on the real dataset at 13 ms per image while the baseline[2] yields the same accuracies and an average latency over 2.5 seconds (s) per image for both datasets.

	mAP (all)	AP_{50} (all)	AP_{75} (all)	mAP (small)	mAP (medium)	AR_{50} (all)	AR_{70} (all)	AR_{80} (all)	AR_{90} (all)	Latency (ms)	Resolution (px)
Mask R-CNN[3]	.466	.985	.317	.340	.489	.990	.740	.279	.023	94.8	448×448
YOLOv4[4]	.882	**.990**	.989	.815	.897	**1.**	**1.**	.995	.873	40.5	320×320
Ours	**.937**	**.990**	**.990**	**.903**	**.945**	**1.**	**1.**	**1.**	**.972**	**16.0**	400×400

Table 1. Average Precision for Max Detection of 100 and Average Recall for Max Detection of 10 computed using MS COCO API.

	Muenster Dataset				ArTe Lab Dataset			
	DR	Precision	Recall	mIoU	DR	Precision	Recall	mIoU
Creusot et al.[5]	.982	-	-	-	.989	-	-	-
Hansen et al.[6]	.991	-	-	.873	.926	-	-	.816
Namane et al.[7]	.966	-	-	.882	.930	-	-	.860
Zharkov et al.[8]	.980	.777	.990	.842	.989	.814	.995	.819
ours	**1.**	**.984**	**1.**	**.921**	**1.**	**.974**	**1.**	**.934**

Table 2. Mean IoU (mIoU), Precison and Recall and Detection Rate (DR) at IoU threshold of 0.5 (Muenster and ArTe-Lab Dataset).

For Y-Net, we use the Microsoft (MS) COCO API, and Pixel-wise metrics to evaluate against Mask R-CNN[3] and, YOLOv4.[4] By default, the MS COCO API configuration evaluates on *small, medium* and *large* areas objects but in our application, the largest detected barcode area is *medium*. Since Y-Net is a segmentation network and does not output confidence scores for each segmented barcode, we propose using *pseudo scores*, the

	Px Acc	Px mIoU	Px Prec	Px Rec
Mask R-CNN[3]	.993	.990	.989	.890
Ours	**1.**	**1.**	**.999**	**.999**

Table 3. Pixel-wise Metrics

ratio of the total number of nonzero pixels in a predicted mask to the total number of nonzero pixels in the corresponding ground truth mask at the location of a given object.

Table 1 shows mAP and mAR values of the models on the synthetic dataset. As seen, our pipeline outperforms Mask R-CNN,[3] and YOLOv4[4] by a mAP of 47.1% and 5.5% and AP_{75} of 67.3% and 0.1% respectively. Also shown in Table 1, is a mAR_{90} improvement of 94.9% and 9.9% on Mask R-CNN[3] and, YOLOv4[4] respectively which highlights that Y-Net continues to yield better mAR results even at higher IoU thresholds. Both our approach and YOLOv4[4] achieve an AR_{50} of 100% and outperform Mask R-CNN[3] by 1%. For *small* area barcodes, Y-Net outperforms Mask R-CNN[3] and YOLOv4[4] by a mAP of 56.3% and 8.8% and for *medium* area barcodes, Y-Net displays a mAP increase of 45.6% and 4.8% on Mask R-CNN[3] and YOLOv4[4] respectively. In addition, Table 3 reveals that Y-Net a has much better semantic segmentation performance than Mask R-CNN.[3] Table 1 displays that Y-Net performs at least 2.5× faster than the fastest of models given by Mask R-CNN[3] and, YOLOv4[4] on LR images.

Similarly, we have used the Detection Rate (DR), mIoU, Precision, and Recall, as described in Namane et al.,[7] Creusot et al.,[5] Hansen et al.,[6] and Zarkhov et al.[8] on the Arte-Lab and Muenster datasets and as can be seen in Table 2, our method outperforms previous works on all of the mentioned metrics. This indicates that our bounding box extraction algorithm is working as expected to detect accurate bounding boxes. However, while it is successful in separating barcodes that are relatively close to each other, it has limitations when barcodes are overlapping. For those occlusion scenarios, the algorithm tends to group the overlapping barcodes into one bounding box instead of separate bounding boxes.

4. CONCLUSION AND FUTURE WORK

In this work, we showed that barcodes can be efficiently, accurately, and speedily detected using Y-Net on UHR images. With *pseudo scores* as *confidence scores*, our approach outperforms existing detection pipelines with a much better latency. In future work, we aim to extend this method to the multi-class detection task for small objects in UHR images and videos in a weakly supervised fashion.

REFERENCES

[1] Mohammed, A., Yildirim, S., Farup, I., Pedersen, M., and Øistein Hovde, "Y-net: A deep convolutional neural network for polyp detection," (2018).

[2] Ren, S., He, K., Girshick, R., and Sun, J., "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **39**(6), 1137–1149 (2017).

[3] K. He, G. Gkioxari, P. D. and Girshick, R., "Mask R-CNN," in [*2017 IEEE International Conference on Computer Vision (ICCV)*], 2980–2988 (2017).

[4] Bochkovskiy, A., Wang, C., and Liao, H. M., "YOLOv4: Optimal speed and accuracy of object detection," (2020).

[5] Creusot, C. and Munawar, A., "Low-computation egocentric barcode detector for the blind," in [*2016 IEEE International Conference on Image Processing (ICIP)*], 2856–2860 (2016).

[6] Hansen, D. K., Nasrollahi, K., Rasmussen, C. B., , and Moeslund, T. B., "Real-time barcode detection and classification using deep learning.," *IJCCI* **1**, 321–327 (2017).

[7] Namane, A. and Arezki, M., "Fast real time 1d barcode detection from webcam images using the bars detection method," in [*Proceedings of the World Congress on Engineering (WCE)*], **1** (2017).

[8] Zharkov, A. and Zagaynov, I., "Universal barcode detector via semantic segmentation," *2019 International Conference on Document Analysis and Recognition (ICDAR)* , 837–843 (2019).