CS230

Build Robust Instance Segmentation for Tennis Court Environment

Application Project - Computer Vision
Project TA: Allan Zhou
Code: https://github.com/yuanzhedong/cs230-mrcnn-ball-detector

Yuanzhe Dong
SUID — 06640893
yzd@stanford.edu

1 Introduction

Manually picking up tennis balls while practicing on the tennis court could be time consuming and tedious. An autonomous tennis ball collector could be a solution. As a first step, we want to build robust instance segmentation model to detect tennis balls, nets, lanes and possible obstacles, using deep learning models Mask R-CNN [1][2]. We will take data-centric AI [3] approach, basically we fix the model architecture and iterate on data. In order to do that we will also build a data collection robot in order to collect new data efficiently. In this project we build a data collection pipeline from scrach and train a Mask R-CNN model to detect balls and nets. The best model we are able to get box mAP: 0.855 and seg mAP: 0.838. The demo video of model prediction is available at https://www.youtube.com/watch?v=q-XErbIsaHA

2 Challenges and Approaches

In order to build a robust model, we're facing several changelings. First traditional Computer Vision approach seems to be cost less due to we don't need data collection or GPUs but turns out it is too sensitive to environmental changes, traditional computer vision method based on the HSV(hue, saturation and value) color space[4], this solution only works if the lighting condition is consistent, and there's no other object in the view shares the same color with tennis balls. Here's an example of using HSV color detection. We covert the first image in figure 1 into HSV color space, then based on the HSV values from the center pixel of the tennis ball and add a threshold to get the bound, here the lower bound being used are [26,81,140] and upper bound [46,101,220]. We then mask out all the pixels with values that are not within the bounds, as showed in the third image in 1. In figure 1 we can perfectly detect the ball as showed in the mask, since we calulate HSV color range based on that image.

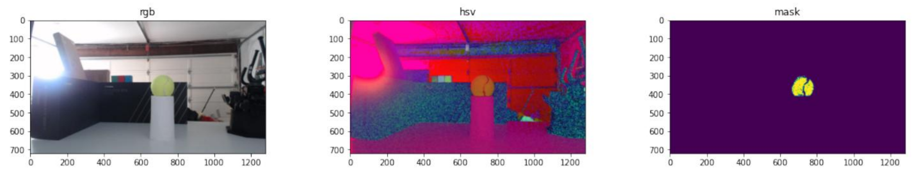


Figure 1: Use this image to detect the HSV upper and lower bounds

However, once we change the environment like the lightening, then it will fail easily, as showed in figure 2. What even worse is we can easily add noise like a bottle with similar color, and the detector will generate false positive predictions, as showed in figure 3

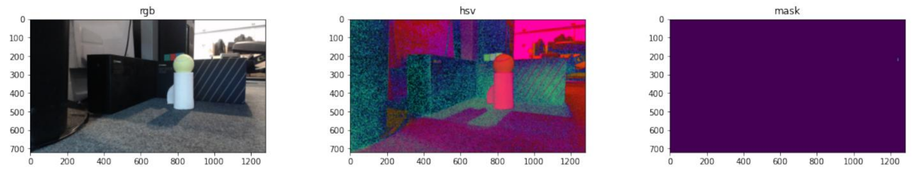


Figure 2: Apply the same HSV bounds to different background and lighting environment

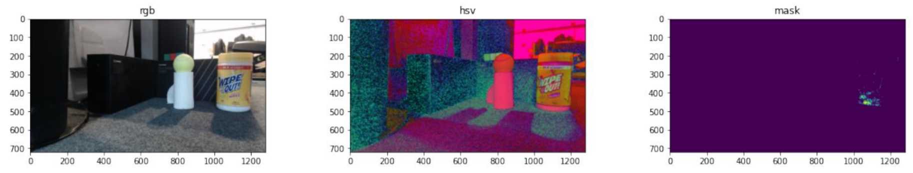


Figure 3: Add noise object with similar color with tennis ball

2.1 Model Architecture

In the end we have to go with Deep Learning approach, specifically we choose Mask R-CNN model. The reason is it will provide fine-grained vision features like instance level segmentations, classes and bounding boxes, and the segmentation maps and bounding boxes could help with not only the ball detection but also further downstream tasks like obstacle avoidance.

Instead of training Mask R-CNN from scratch, which may need a lot of labeled data and more time to converge, we use maskrcnn with resnet50[5] backbone model pre-trained on COCO dataset[6], and then fine-tune it using our own dataset. More specifically we use pre-trained model weights to initialize our model except box head and mask head as shown in figure 4. For box head and mask head using Kaiming normal initialization.[7]

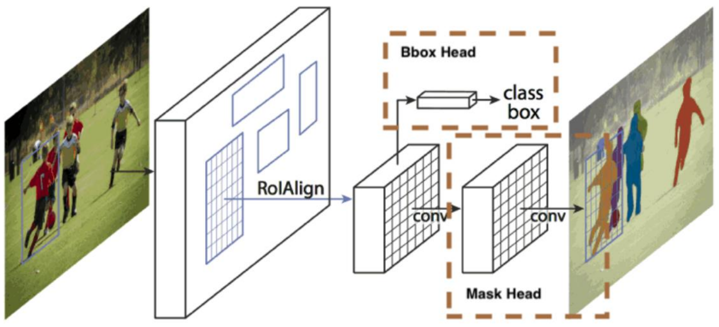
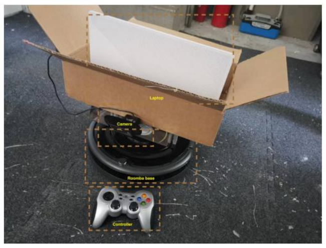


Figure 4: Finuetune mask head and box head

The second challenge is we need to build the data collection pipeline from scratch, including hardware setup, data collection and data labeling.

2.2 Hardware Setup

We build a mobile robot with camera to collect the data and label them from scratch. The robot built for data collection includes a Roomba base, a logitec camera, a laptop and a joystick to control the Roomba base. We randomly put the balls and use the joystick to control the robot base to move around and collect raw frames.



(a) Robot Hardware Setup

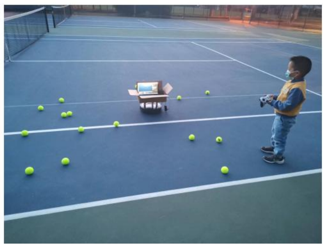

(b) Data Collection from tennis court

Figure 5: Data collection

2.3 Data Collection

We find a tennis court near our home and then randomly throw tennis balls on it. We then use the controller to control the robot and move around randomly. Raw videos are stored on the laptop for further use.

2.4 Data Labeling

The are several open sourced labeling tools available, the one we choose is labeled using labelme [8]. It supports labeling circles and polygon, we use circle for tennis ball and polygon for net.

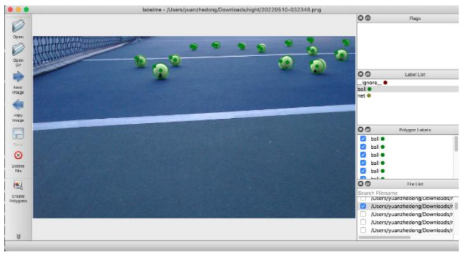
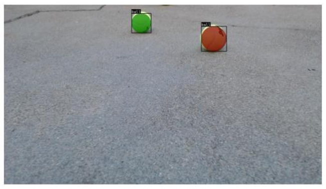


Figure 6: Label data using Labelme

3 Dataset

We have launched 5 rounds of data collection on 4 different dates, for each round we collect about 15 min raw videos. Data from the first three rounds are being splitted into frames, then some of them being labeled and used for model training and metric evaluation. The last two rounds are used for qualitative research.

	Round 1	Round 2	Round 3	Round 4	Round 5
Location	Backyard	Tennis Court	Tennis Court	Backyard	Tennis Court
Time	Evening	Evening	Morning	Morning	Morning


(a) Sample labeled data from our backyard

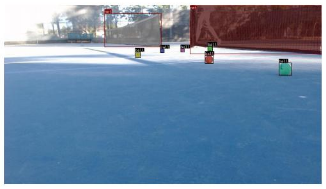

(b) Sample labeled data from real tennis court

Figure 7: Sample labeled data from the view of Robot with label

We build one dataset after first two rounds and then build another dataset after round 3. We convert the videos to frames firs then random choose some frames to label, then we split the labeled frames into training set and test set. Each fram has it's timestamp to indicate when its being collected, frames being collected earlier will go into training set and the later ones will go into test set. So there's no overlaps. The details counts of the datasets are showed below:

	Evening	Evening + Morning
train	22	139
test	13	35

3.1 Data Augmentation

We also applied data augmentation, we will randomly flip the image and labels as well as apply blurring, rgb shit, random shadow and random sub flare etc. Details are in the code.

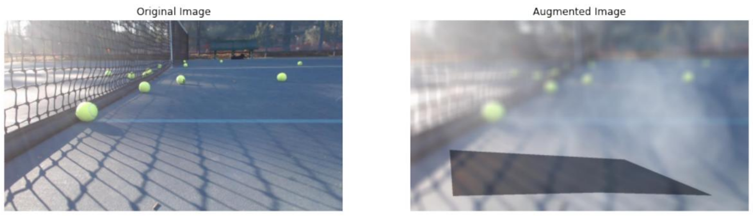


Figure 8: Input image before and after data augmentation

4 Evaluation

Due to we have a small set of labeled data, instead of building a dev set, we pick the model that has lowest loss from training set and report the box mAP and segm mAP[9] on test set. The IoU threshold we're using the 0.75. We use the video data from last two rounds evaluate possible overfitting issue qualitatively.

	Evening	Evening + Morning	Evening + Morning + Data Aug
Training set Loss	0.353	0.377	0.480
Test set Box mAP_{75}	0.528	0.855	0.822
Test set Seg mAP_{75}	0.493	0.838	0.804

According to the metric the model trained without data augmentation is better. The visualisation of the predictions also indicate that. With data augmentation, the model missed one image as shown in figure 9 (second row first column), while it works for the model trained without data augmentation as shown in figure 10(fourth row fifth column). There could be two reasons that data augmentation doesn't help here. First reason could be the data augmentation we're using is too aggressive and it confuses the model. Second the variance of the testset is still not large so it can't reflect that data augmentation can help with the model generalization.

5 Conclusion and Future work

In this project we are able to build the end to end hardware and software system to build a robust vision model to detect tennis ball and net from tennis court. The demo video is available at https://www.youtube.com/watch?v=q-XErbIsaHA. The code is available at https://github.com/yuanzhedong/cs230-mrcnn-ball-detector.

For future work, first we need to label more data. Currently we only have one person to label the data and we plan to cross-sourcing like Amazon Mechanical Turk to scale our data labeling. Secondly we plan to deploy our model to a device with lower power consumption instead of the RTX3060 we're using right now, so the robot can run for a longer time before recharging.

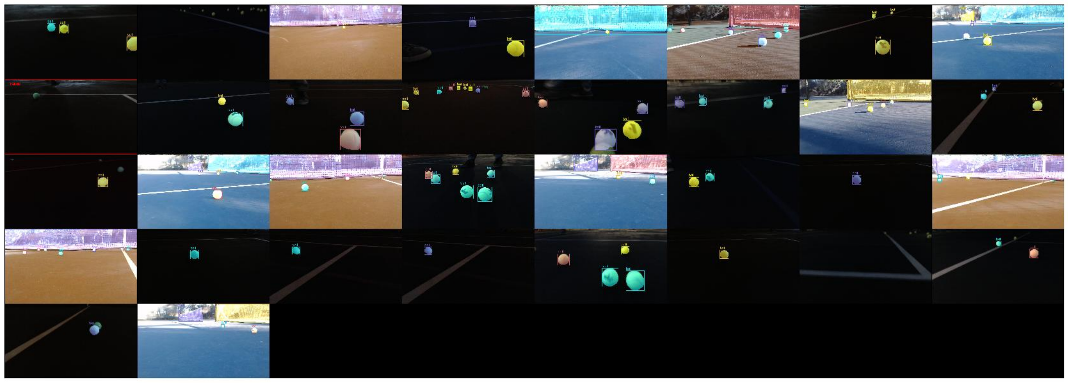


Figure 9: Visualization of model prediction for test set with data augmentation

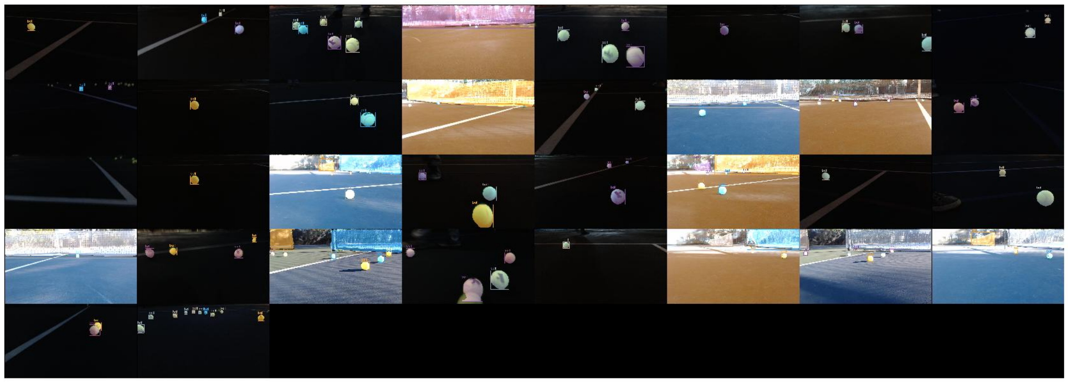


Figure 10: Visualization of model prediction for test set without data augmentation

6 Acknowledgements

Thanks to my son who helped to collect the data as shown in figure 5b. And thanks to TA Allan Zhou for his guidance, really appreciate his patient to answer all my questions. Also thanks to all the TAs and instructors of CS230, I've learned a lot from this course.

References

[1] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn, 2017.

[2] Lilian Weng. Object detection for dummies part 3: R-cnn family. *lilianweng.github.io*, 2017.

[3] Andrew Ng. https://landing.ai/data-centric-ai/.

[4] S. Sural, Gang Qian, and S. Pramanik. Segmentation and histogram generation using the hsv color space for image retrieval, 2002.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[6] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2014.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

[8] labelme. https://github.com/wkentaro/labelme.

[9] AP. https://towardsdatascience.com/a-better-map-for-object-detection-32662767d424.