AML homework 2

Introduction

There are 3 different datasets that were created for this homework:

1. Dataset that I manually annotated from raw old pictures
2. Dataset that was annotated automatically using Grounded DINO from raw images (and then applying SAM for segmentating)
3. Dataset that was automatically converted from founding boxes from the HW1 to segmentation. (with a help of SAM)

I provide 2 jupiters notebooks for training the first Dataset (annotated by hand)just for reference, the main solution - jupiter notebooks that were trained on the 3rd dataset

Step 1: Taking pictures

For this Assignment I simply used the previous pictures. Nothing new

Step 2: Annotating

First dataset

As I said in introduction - there are different sets that were annotated. For the manual one - I decided to classify 4 types of items in vending machines. Simply repeated the procedure of cliking on the needed item, fixing the highlighted area by adding new points and then choose an appropriate class.

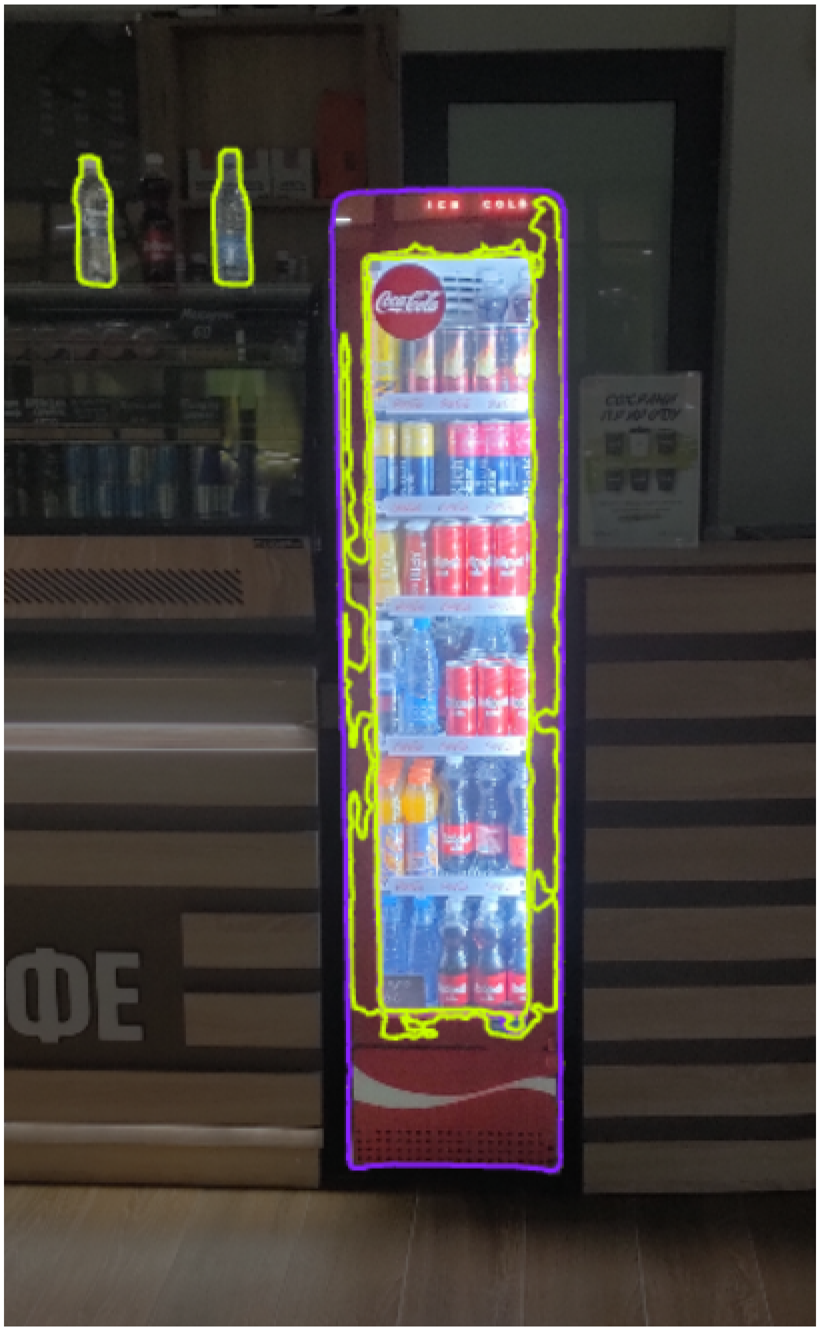
In general, nothing complicated, but monotonous and boring process

Second dataset

I found that on 21st of April Roboflow uploaded a very helpful article on how to aitomatically annotate your dataset for segmentation. So I checked it out (even after I already trained the models for the 1st dataset).

Second dataset was annotated from raw data with help of grounded DINO model, so it has some nuances.

- You have to provide names for your classes, so the DINO model could find objects of interests. Because of that it’s hard to annotate something really specific like 2 different types of corals. So you can use only some general words like “dog”, “person”, “shoe” e.t.c
- Even for small thresholds the model struggles to find some objects, so you will have to add some objects manually
- Low thresholds also implies that sometimes model will incorrectly highlite something wrong, in my case it highlighted the whole vending machine as an object. However this types of problem can be solved simply rejecting all image areas that are bigger than some other threshold, that you choose. But potentially it could lead to incorrect interpretations.



- For the same object there may be several classes. It may bring a problem when you have a class that is a subclass of another or has some intersection. In my case aluminium cans of soda are both a drink and a can(which is an intersection of 2 classes), but I wanted to treat them differently.

Overall I was satisfied with the speed of annotating data (and the ML approach to the problem), but the images didn’t have a lot of class instances, so there was not enough data for training model. Therefore I came to the 3rd and final dataset.

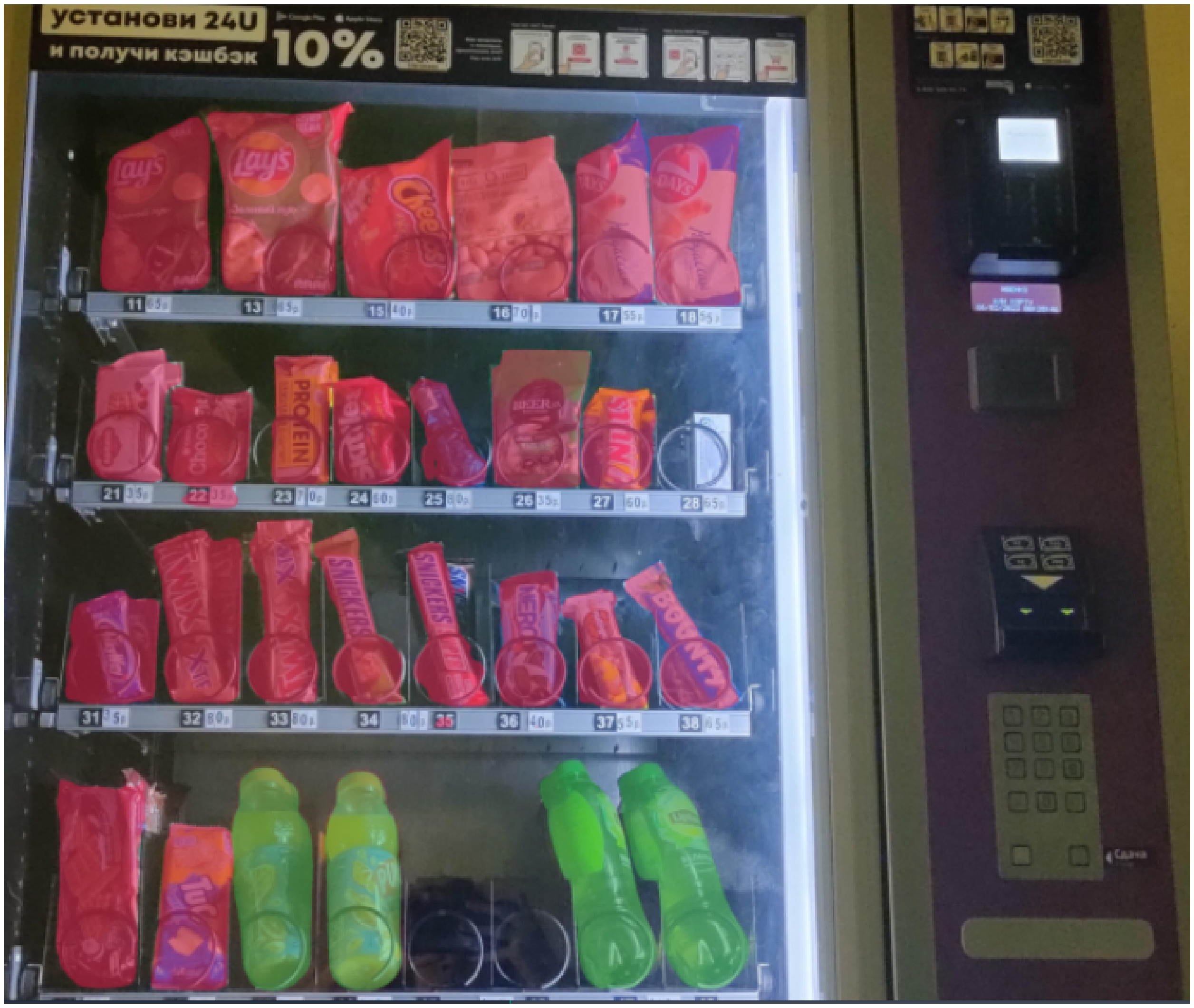
Third dataset

This dataset was made in the same jupiters notebook.

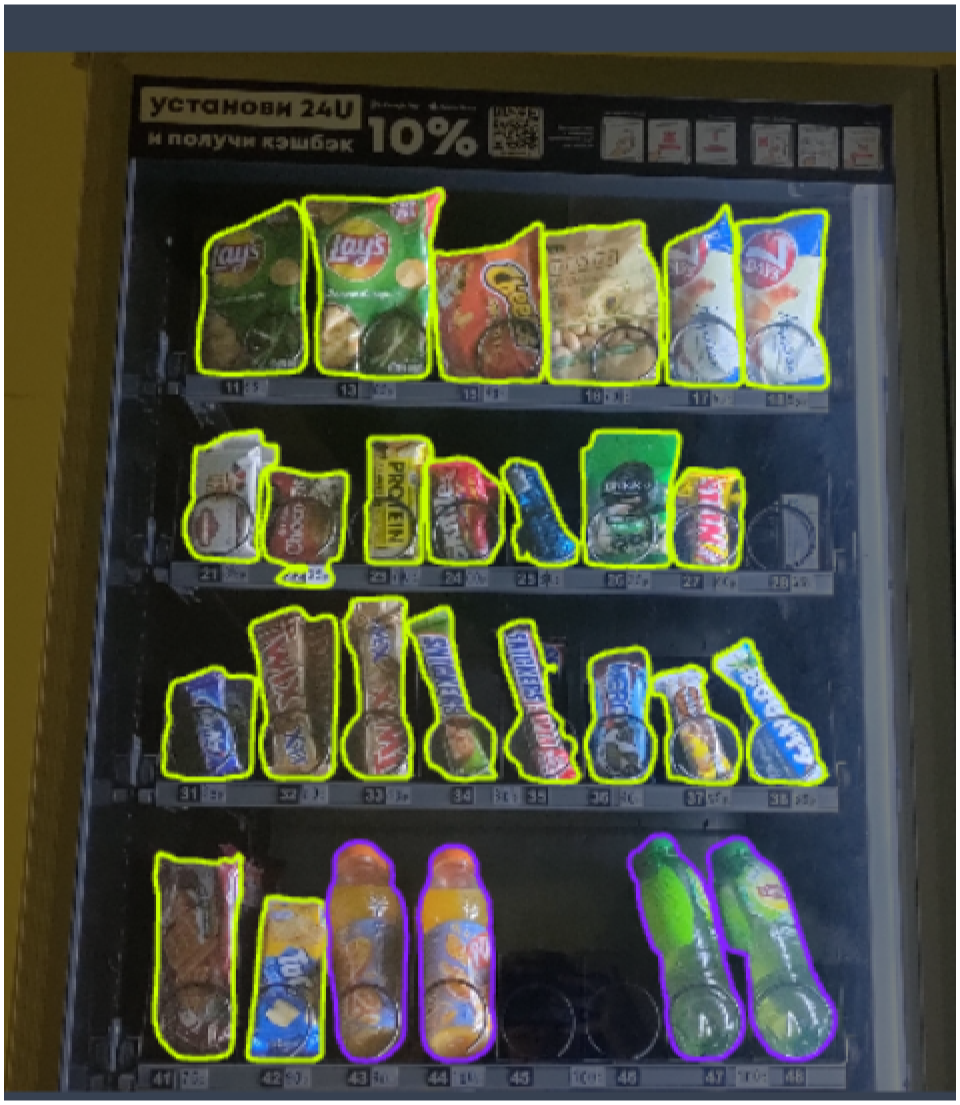
I just accessed my dataset from HW1 and used SAM for converting finding boxes to segmentation polygons.

Note that in the HW1 I had just 2 classes: food and drink, in contrast with 2 previous datasets

Picture of an annotated image in jupiters notebook



Picture of the same annotated image in roboflow



There are some problems with annotations - sometimes a polygon is larger then a real object, so for improvement of training models I could adjust that. However I overall satisfied with them and do not want to spend more time on that - therefore I leave it as it is.

Example of the problem (I zoomed picture)

Step 3: training a Mask RCNN model using detectron2

Here it’s as simple as in the first assignment. I adjusted some parameters like Patience for early-stopping and overall number of epochs, so I don’t have to vait too much as I trained Masc RCNN for the first dataset, which took almost an hour in google collab on GPU.

The .ipunb file can be found in the github

Step 4: Train Yolov8 the smallest size for segmentation

The smallest size of yolov8 for segmentation - nano. So I use yolov8n-seg for the 3rd dataset.

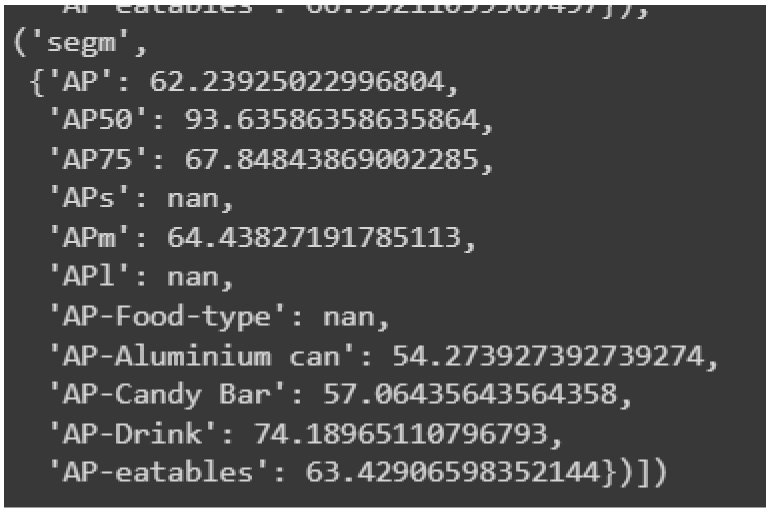
On the first dataset I did not succed to train small model on small number of epochs, so I tried to use the largest for 50 epoch which gave me some results (before that it didn’t predict anything).

For the final solution (3rd dataset) I tried to train the smallest model on epochs and left it as it is

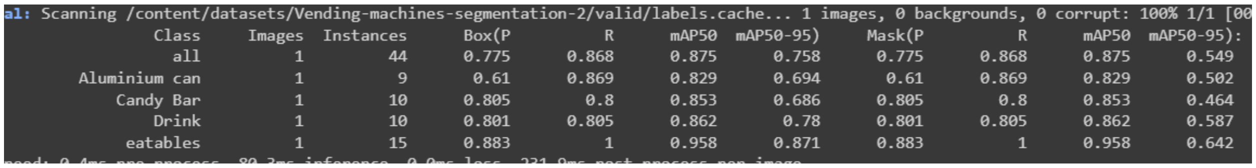
Step 5: Comparison

First comparison in time : I spend almost an hour to train the Mask RCNN in comparison with for yolov8x

Second - in size: yolov8n is definitely smaller than Mask RCNN, but I’m not sure with yolov8x mAP: data for Mask RCNN (1st dataset currently)



```
('segm',
 {'AP': 62.23925022996804,
  'AP50': 93.63586358635864,
  'AP75': 67.84843869002285,
  'APs': nan,
  'APm': 64.43827191785113,
  'APl': nan,
  'AP-Food-type': nan,
  'AP-Aluminium can': 54.273927392739274,
  'AP-Candy Bar': 57.06435643564358,
  'AP-Drink': 74.18965110796793,
  'AP-eatables': 63.42906598352144})])
```



```
al: Scanning /content/datasets/Vending-machines-segmentation-2/valid/labels.cache... 1 images, 0 backgrounds, 0 corrupt: 100% 1/1 [00
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95)     Mask(P          R      mAP50  mAP50-95):
                   all          1         44      0.775      0.868      0.875      0.758      0.775      0.868      0.875      0.549
         Aluminium can          1          9       0.61      0.869      0.829      0.694       0.61      0.869      0.829      0.502
             Candy Bar          1         10      0.805        0.8      0.853      0.686      0.805        0.8      0.853      0.464
                 Drink          1         10      0.801      0.805      0.862       0.78      0.801      0.805      0.862      0.587
              eatables          1         15      0.883          1      0.958      0.871      0.883          1      0.958      0.642
```

Surprisingly yolov8x has a better results compared to detectron2 in terms of average precision for segmentation.

However I would expect a worse, but faster result from yolov8n as it was in the first Assignment snce there is a tradeoff between speed and performance