Comp Aspects of Robotics HW2

Manasi Sharma: ms5510

October 2020

1 Problem 2.1

Instance segmentation is a type of image segmentation (partitioning an image into different portions/segments) where different kinds of objects (apart from the background) are partitioned into different classes, associated with certain colors. In our case, we aim to perform instance segmentation to partition the image into segments corresponding to the bowl, can, banana, and other objects background, associated with numbers from 1 to 5.

The inputs to the model are the RGB images of the various scenes of the objects. The outputs are the masks corresponding to the images of particular scenes, with corresponding pixels having values associated with the object they comprise. For example, all pixels in the background have a value=0, while the banana has a value=4. In this way, the model essentially labels the certain areas of the input image with a tag associated with the object, to group areas of the image as belonging to one object.

We use fully supervised learning for this problem (where we/humans have provided all of the labelled/target data for training). This may be more appropriate as the mask "prediction" is more than a single label and generating it through weak supervision can weaken our learned model's performance. Also our training dataset (n= 300) is small enough to provide labelled data for.

However, for the last problem, we use our predicted masks to make pose estimations, which involves weak supervision (since one of the elements used for training is generated by another model, as opposed to labelled by a human).

2 Problem 2.4

(1) In most convolutional layers, our goal is to incorporate data from the surrounding pixels in order to extract features (by applying some operation). For this purpose, the output for a single convolution operation encapsulates information from more than one pixel. It also reduces the size of the resulting image

by a bit. After this layer, the maxpool and interpolation layers help to either down sample or add these features together.

However, right before output, our goal is to not to incorporate more information from the surrounding pixels; therefore the 1x1 goes through each pixel individually, and does not change the resulting dimensions of the image but rather reduces the number of channels in the image (which help us to make a certain output). For example, in our model, it does not apply a unique convolution operation, but rather reduces the number of channels from 16 to 6, to fit our output purposes.

(2) In the mask, each pixel stores a value associated with the "label" of that pixel (which object it corresponds to). Eg. 0 for background and 4 for banana. In the output for our model, this value is represented in 'one-hot encoding', where each of the 6 channels is a placeholder for the 'label' and is filled with a value corresponding to the probability of that label. The index of the channel with the highest label would represent the value for the whole pixel, and through a simple max() operation, this would correspond with the 1-channel ground-truth mask.

3 Problem 2.5

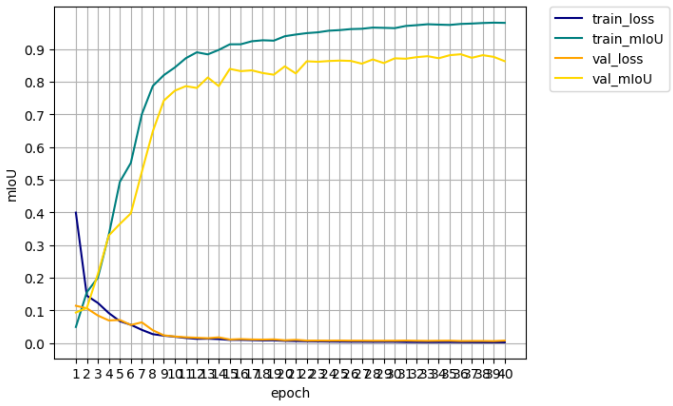

Figure 1: Learning curve plot

Note: I came very very close to hitting the 0.90 mIoU mark but was not able to even after training for many epochs, and with many different variations of batch size and learning rate (after training for hours on GCP). However, I came very close (0.88) and my final predictions in Q3 were relatively good (coincided spatially) for the validation set, so I hope you will consider giving me credit for

this.

4 Problem 3.2

1. minimum change in cost (threshold)
As I lowered the threshold, (or the minimum change in cost, above which certain pairs of points are rejected from consideration), the "Average Closest Point Distance" output reduced by a bit. For example, when I changed the threshold from 1e-05 to 1e-06, the avg. closest point distance ("output") reduced by 0.00002-3 for both the predicted pose using ground truth mask and the predicted pose using the predicted mask. [I did notice though that some objects, such as the banana and the bowl, had less improvement than the others]. Reducing to 1e-7 reduced the avg. distances by anywhere between 0.00005-0.0001, but reducing the threshold to 1e-08 did not change much.
As I increased the threshold (to 1e-04 for eg.), the results were somewhat consistent, but the distance increased on average– some objects had their outputs almost double, while others increased by 0.0001.

Therefore the trend seemed to be as the threshold decreased, the avg. closest point distance also decreased. I found the optimal threshold value to be 1e-07.

2. maximum number of iterations
As I increased the number of iterations from 20 to 30 and then to 40, the avg. closest point distance reduced by a small amount, 0.0001-2. As I increased it further to 50 and 60, the output became more inconsistent across the objects (for some, the output increased slightly) and did not reduce significantly more than 0.00002.

Therefore the trend seemed to be as I increased the maximum number of iterations as the matrix is refined over each iteration of the algorithm. I found the optimal $max_iterations$ value to be 40.

3. initial transformation
When I tried using a 4x4 Identity matrix instead of the initial matrix provided by the Procrustes function, the avg. closest point distance was 1e-05 units larger than the output from the Procrustes function (for objects with predicted pose using ground truth mask) and almost a factor of 10 larger for scenes evaluated using predicted pose using predicted mask.
Therefore, an accurate initial matrix (as opposed to a random one) improves the performance of the ICP algorithm.

5 Extra

1. Denoising
The function is implemented in icp.py, and I have included an extra function in

$segmentation_helper.py$ to isolate the mask for a single object (eg. green for the tomato soup can).

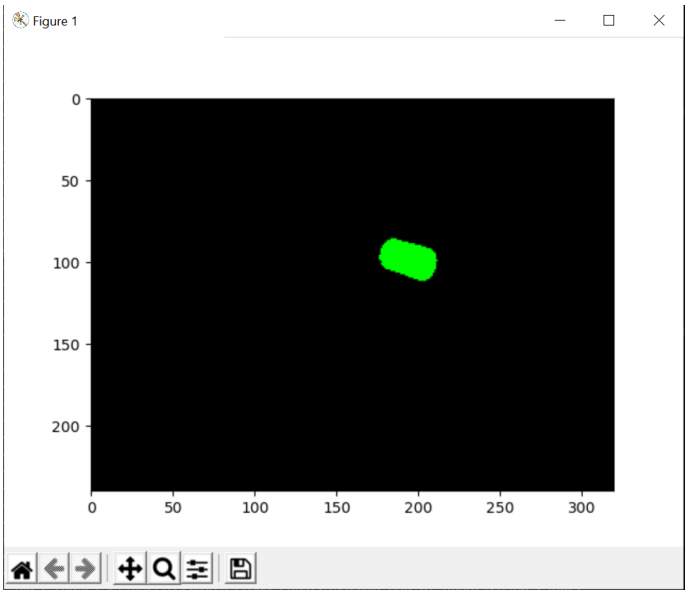

Figure 2: Original mask with no denoising

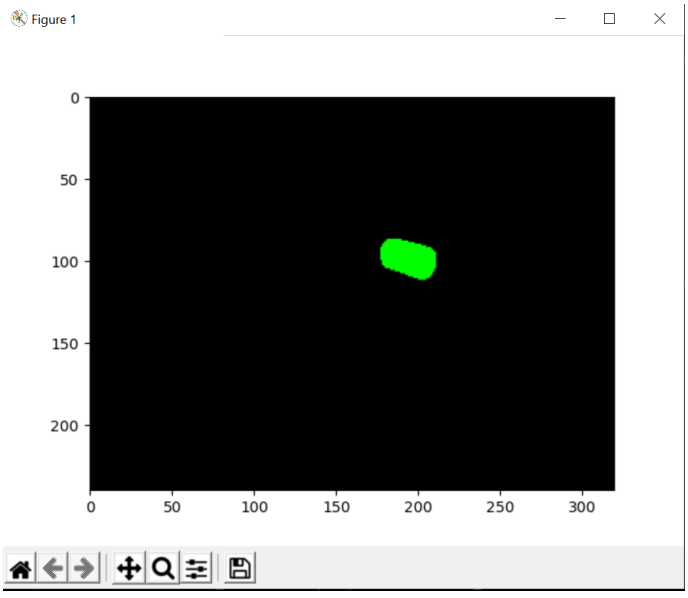

Figure 3: Denoised mask

2. Pose Estimation

I am unsatisfied with $4_rgb.png$ in the test set, as the predicted point cloud does not quite match with the rgb image (many objects are clustered together in a bit of a mish-mash). No matter how much I decreased the threshold or increased the number of $max_iterations$, the output for the banana did not improve much (and often increased when not expected), and I believe that when the banana is behind other objects in the scene it is difficult to segment using ICP (since many points are inaccessible). In this scene, the banana (and other objects) are clustered together in one segment of the image, so I believe the prediction for this scene did not do as well as the others.