COMS4733 Qianhuang Li ql2465

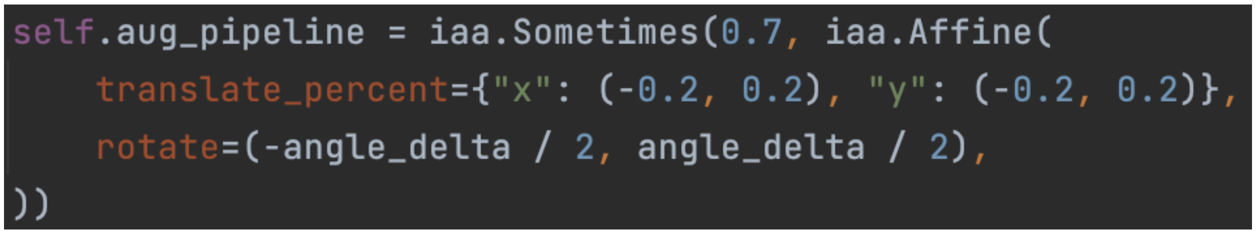
Homework 4

2a

Compared with the one-hot pixel image, the images using Gaussian Scoremap indicate not only a center grasp point but also the pixels around the best one. As each object can have more than one affordance, the Gaussian map will give out the most promising point with other low-possibility points instead of just one 100% point. And it is easier to train and converge as well.

2b

In your report, **explain** in English what transformations are done in *self.aug_pipeline*.

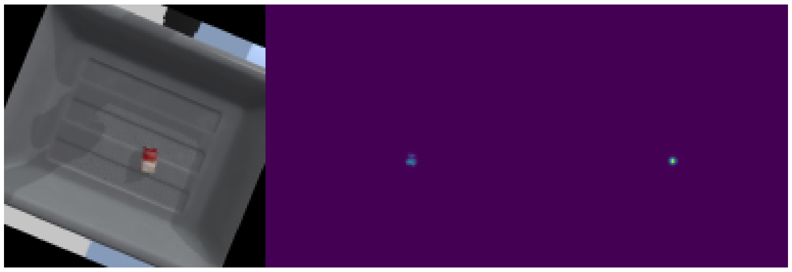


```
self.aug_pipeline = iaa.Sometimes(0.7, iaa.Affine(
    translate_percent={"x": (-0.2, 0.2), "y": (-0.2, 0.2)},
    rotate=(-angle_delta / 2, angle_delta / 2),
))
```

Self.aug_pipeline applies the affine transformation to 70% of the RGB images, in which translate/move the x/y axis by -20 to 20, rotate in range [-22.5/2, 22.5/2].
This will make the prediction more robust by adding more variances to the existing images.

2d

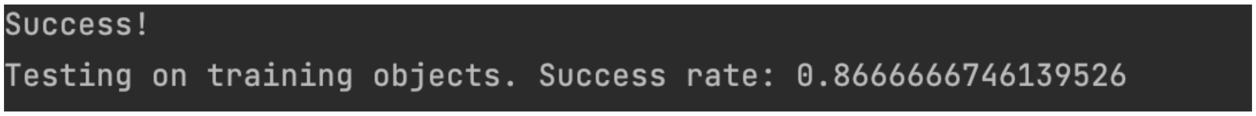
```
step# 273 training loss 0.001130242133513093
Epoch ( 40 / 101 )
---------------------------------
Train loss: 0.0011
Test loss: 0.0010
---------------------------------
checkpoint saved at epoch 40
Saving predictions in directory data/affordance/training_vis
```

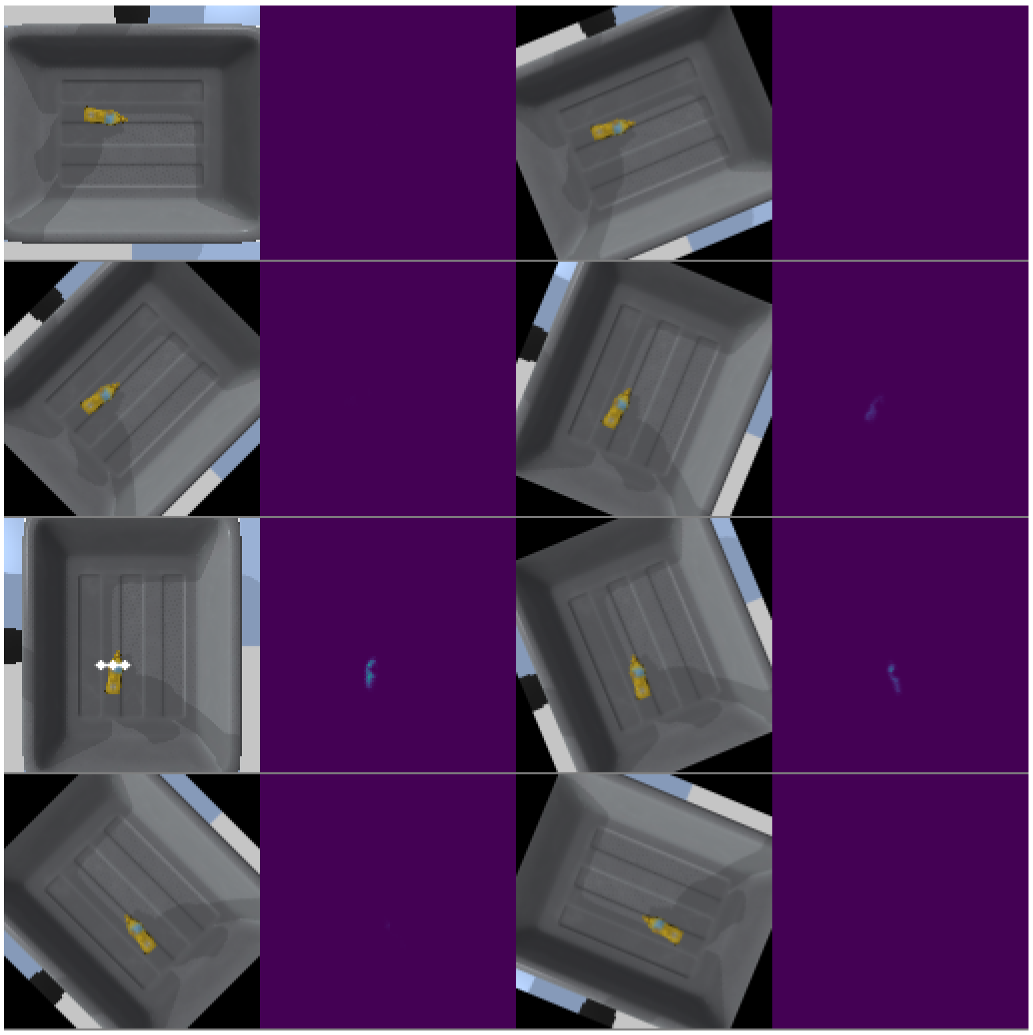


2e

Video Link

success rate: 0.86
YcbMustardBottle

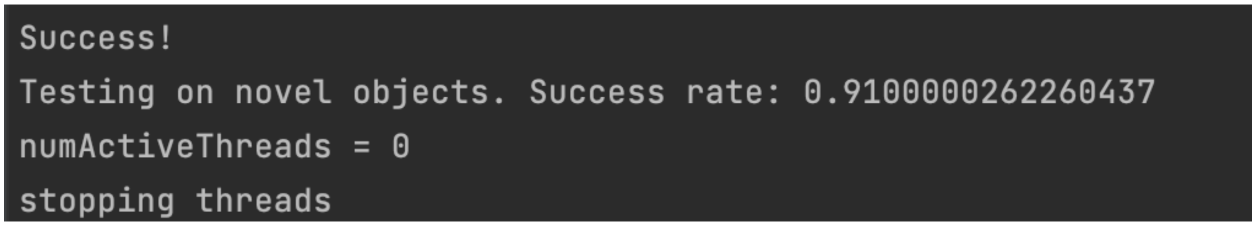

```
Success!
Testing on training objects. Success rate: 0.8666666746139526
```



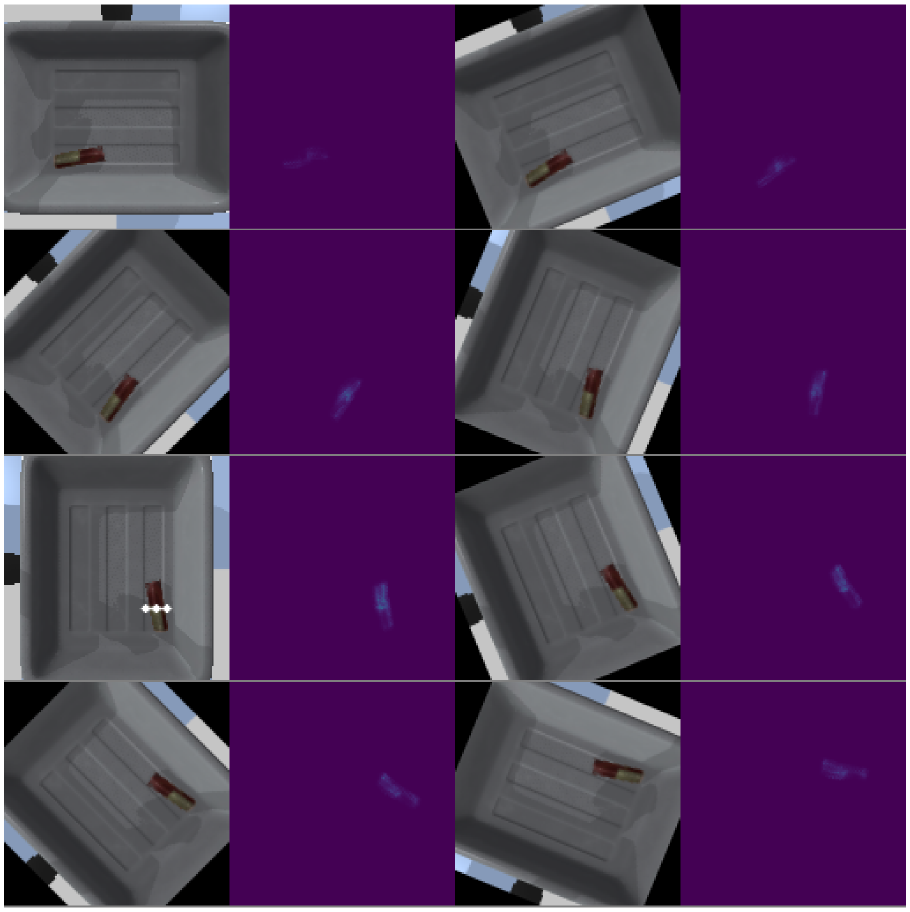
2g

Video Link

success rate：0.91



```
Success!
Testing on novel objects. Success rate: 0.9100000262260437
numActiveThreads = 0
stopping threads
```

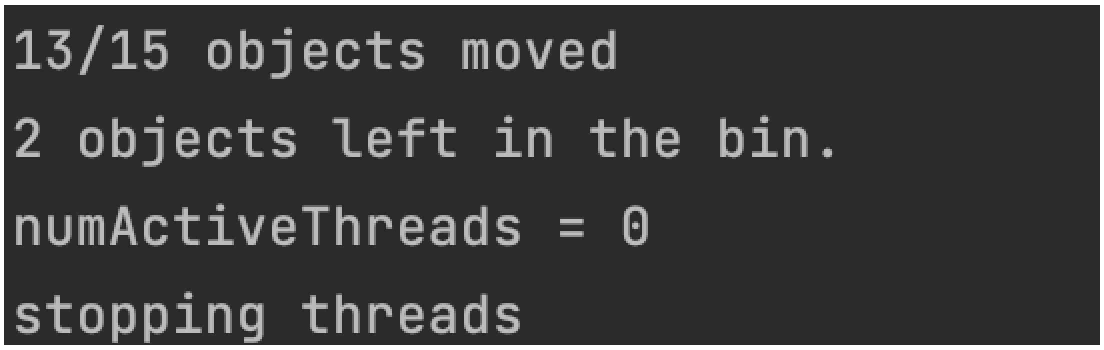


Describe how the image might be different or similar to the visualizations from the previous part：
Most of the objects were predicted well enough and both the RGB images and prediction images are shown similarly as well.
The grasp position shown in the visualization looks close to the similar shape in train objects.
For the ChefCan, the gripper kept on reaching the edge of the bin. One guess is that the color merged into the bin and the shadow. Or maybe the edge looks more like the Clamp.

2h

Video Link

Object left: 2
Seed: 4



```
13/15 objects moved
2 objects left in the bin.
numActiveThreads = 0
stopping threads
```

I have changed seed 2 to seed 4. As when I was trying seed 2, the gripper stuck on the clamp, and was never able to pick it up. Thus it can only pick 5 in total with seed=2.

2i

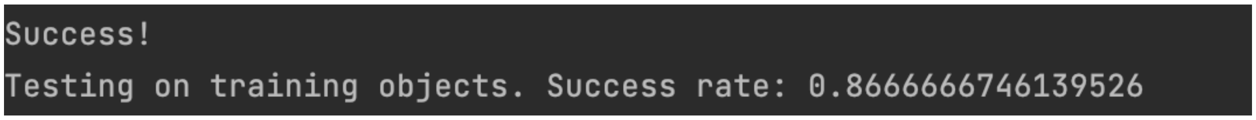
Now that you have evaluated the model, in your report, try to **explain** why is this method is sample efficient, i.e. performs well on both seen and unseen objects despite given only 60 images to train.

The method is sample efficient because of transitional equivalence. It means that the position of the objects should not be fixed in order for the CNN to detect it. We don’t need to feed the CNN all possible images of the objects in all possible positions/angles because the gripper will follow the object, using the Gaussian map to locate the output. Here, all weights are considered the same, meaning that an image and all its transitions are equivalent, so they should have the equivalent translation in outputs. Transitional equivalence improves the CNN in generalization abilities when the data set size is limited.

3d

Success Rate: 0.86

For Training: Video Link

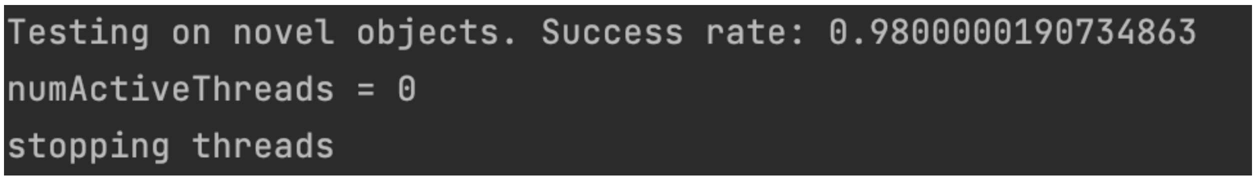


```
Success!
Testing on training objects. Success rate: 0.8666666746139526
```

The success rate for the training set is the same as in part 2. The gripper did change angles and positions for the failed objects, but I assume the first grasp is the best grasp in the training set and the failure is unavoidable. Thus the rate didn't change.

Success Rate: 0.98

For Test: Video Link

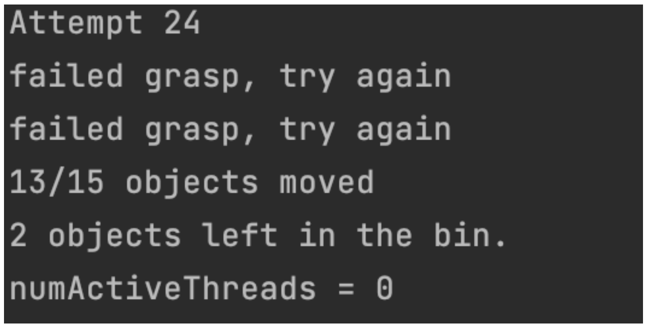


```
Testing on novel objects. Success rate: 0.9800000190734863
numActiveThreads = 0
stopping threads
```

The success rate improved from 93% to 98%. As the buffer is added to the pipeline, the failed grasp will deduct the affordance of the previous prediction map. And for the next try, the gripper chose another position/angle. Therefore, its success rate increased due to several tries.

Object left: 2

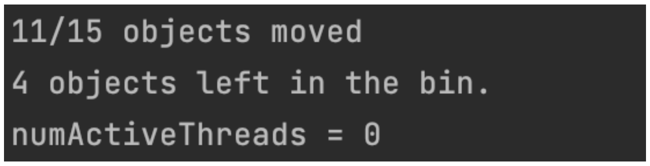
For Empty Bin: Video Link   with Seed: 4



```
Attempt 24
failed grasp, try again
failed grasp, try again
13/15 objects moved
2 objects left in the bin.
numActiveThreads = 0
```

The number of objects picked was not added, as the without buffer one is doing ok already.

But for seed= 2, the action improved a lot with buffer, from picking up 5 to picking up 11.

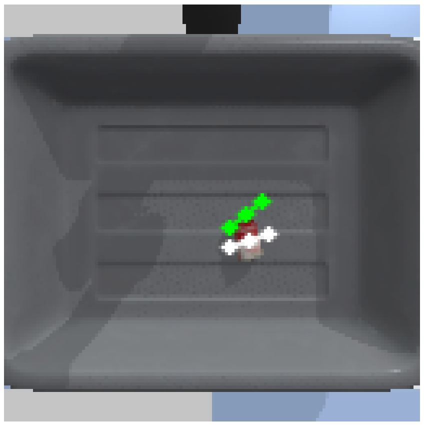


```
11/15 objects moved
4 objects left in the bin.
numActiveThreads = 0
```

4a

report your training loss and test(validation) loss. Include an image from data/action regression/training vis.

```
Epoch ( 98 / 201 )
---------------------------------
Train loss: 0.0132
Test loss: 0.0097
---------------------------------
checkpoint saved at epoch 98
Saving predictions in directory data/action_regression/training_vis
```



4b

[Video Link]

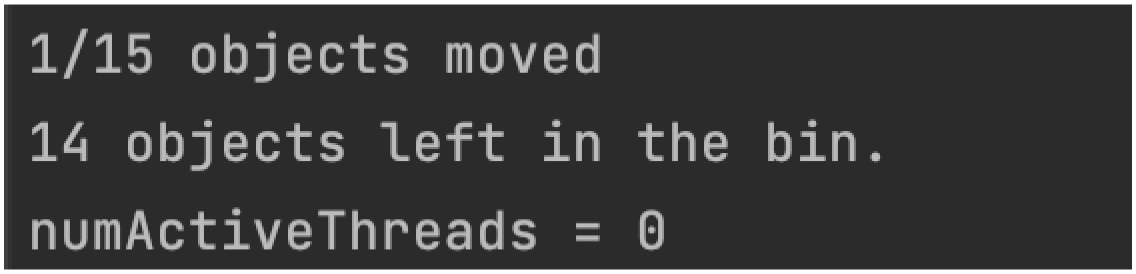
Success Rate: 0.20

```
Testing on training objects. Success rate: 0.20000000298023224
numActiveThreads = 0
stopping threads
```

4c

Video Link

Object left: 1
Seed: 4



```
1/15 objects moved
14 objects left in the bin.
numActiveThreads = 0
```

Also **explain** why this method performs worse compared to Visual Affordance.

The Regression Model performs worse because it fails to evaluate the coordinates and the angles of the objects with regard to their affordance. The regression is more likely to misclassify the objects due to its limited dataset. In inaccurate action, the regression model will give out the same loss for different scenarios, making the output to be also inaccurate.