Introduction to Computer Vision (Spring 2023)
Assignment 4

Release date: May 20, due date: June 3, 2023 11:59 PM

This assignment includes 2 tasks: implementing and training a PointNet for classification and segmentation, and training a Mask R-CNN, which sums up to 100 points and will be counted as 10 points toward your final score for this course. This assignment is fully covered by the course material from Lecture 11.

Submission: To submit your homework, please compress your code **and your results except dataset and checkpoints** following the original path structure, and submit to course.pku.edu.cn. Feel free to post in the discussion panel for any questions and we encourage everyone to report the potential improvements of this assignment with a bonus of up to 5 points.

1. **PointNet (50 points):**

   PointNet is the most widely adopted neural network for point cloud learning. In this question, you are requested to implement the pipeline of PointNet for both classification and segmentation tasks on ShapeNetPart[1] and then visualize the features. Please read and follow the README to prepare the environment and dataset.

   a)[40 points] Points Classification and Segmentation.

   First of all, you need to build the PointNet by following the architecture in Fig.1. In implementation, you will build separate networks for different tasks with different feature dimensions. We provide the off-the-shelf data-loaders of ShapeNetPart for both classification and segmentation tasks. The training process may take you about 20 minutes. The content of this question can be found in *model.py*.

   a.1) **For segmentation task**, your network should predict the part labels of the given point cloud. Specifically, we consider the "airplane" category. Please refer to *train_segmentation.py* for more details.

   a.2) **For classification task**, your network should predict the category of given point clouds. You are expected to investigate the effect of the dimensions of the global feature. So in this classification part, you are requested to train two separate PointNets. One is the original PointNet with $1024D$ global feature. And another one uses $256D$ global feature.

   You will have a similar training curve as Fig.2. You may find that a bigger network with more capacity non-necessarily outperforms a smaller network. Please check *train_classification.py* for more details and submit the screenshot.

   b) [10 points] Point Feature Visualization

[1] Yi L et al. A scalable active framework for region annotation in 3d shape collections[J]. ACM Transactions on Graphics (ToG), 2016

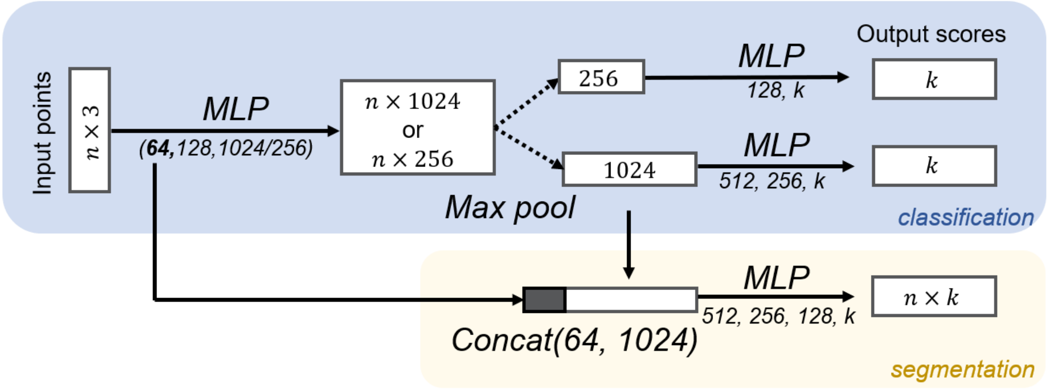

Figure 1: PoinNet architecture for both classification and segmentation tasks.

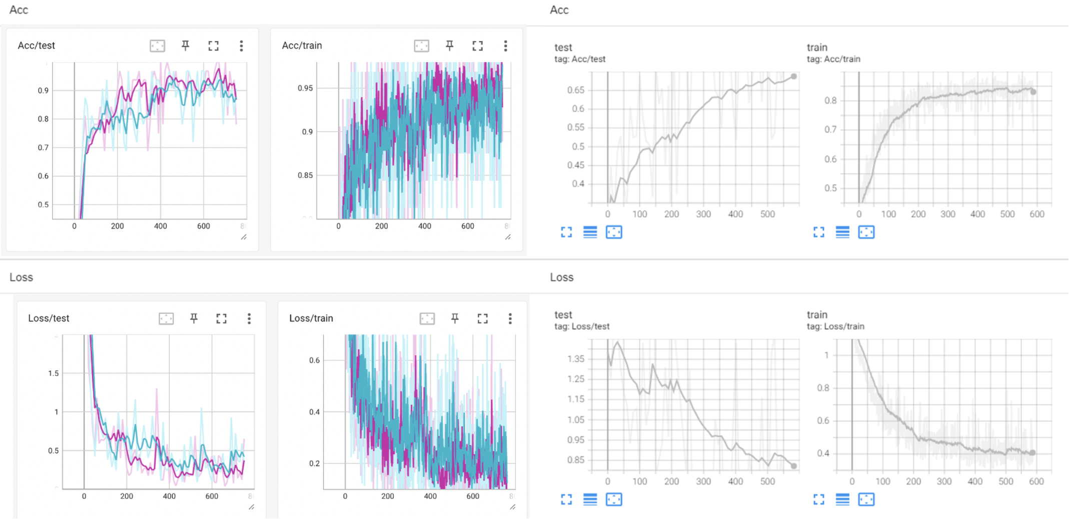

Figure 2: Classification task with 1024D (pink) and 256D (blue) global feature. Segmentation task with 1024D global feature (gray)

We have already trained some networks and draw lots of curves. But sometimes, we want a more comprehensive way to understand what the network actually learned. So in this question, you are required to visualize the *cruciality* of $n \times 1024$ points feature before max-pooling in the classification network(1024D). And the *cruciality* is simply defined as the maximum value along the point feature dimensions. The colormap and points-to-ply functions are provided and you can obtain similar colored point clouds as Fig.3. please see *feature_vis.py* for more details.

2. **Mask RCNN (50 points):**

Mask RCNN is the milestone of the 2D instance segmentation. Even now, it's still considered as one of the most important baselines in the field. In this question, you will have the chance to play with a small Mask RCNN by predicting the instance segmentation of simple shapes, *e.g.* triangular, sphere and rectangle. We build the Mask RCNN with *torchvision*[2] and have replaced the backbone with lightweight mobilenetV2. Some settings have been modified to make it easy to run on pure CPUs

[2] https://pytorch.org/tutorials/intermediate/torchvision_tutorial.html

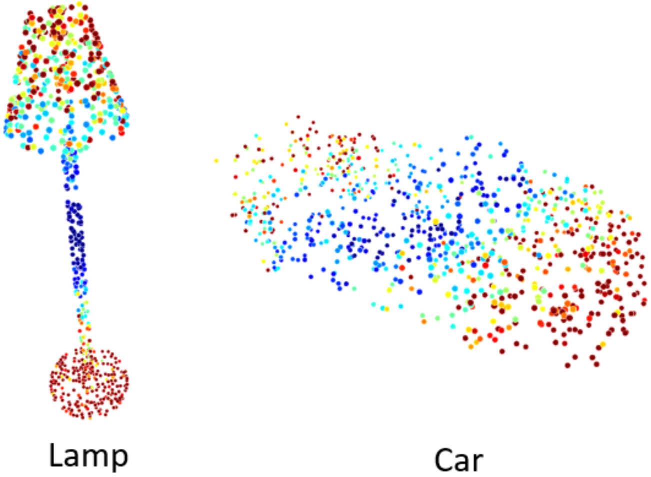

Figure 3: Feature visualization with colormap. Your results may not be exactly the same.

machines. You can create new functions for your convenience. The content of this question can be found in *MaskRCNN*.

a)[20 points] Prepare the dataset

Data preparation is always the first if you want to get a network work on your own data. In this question, you are requested to generate a dataset that contains rectangles, spheres and triangles. Our requirements are:

1. The number, position and size of the shapes should be random. And the color for both shapes and backgrounds should be random.

2. We do not want large occlusions between shapes. So you need to implement a non-max suppression (NMS) to remove the shapes with large overlap.

3. The detailed parameters are listed in *dataset.py*. We also encourage any creative ideas to make the data preparation more interesting.

Several examples can be found in Fig.4. Please complete the *MaskRCNN/dataest.py*.

b)[20 points] Train the Mask RCNN

Now, with all data prepared, we are ready to train a Mask RCNN. You can obtain and load the pre-trained weight to boost your training process by following the README. And the log will be automatically saved in *results/maskrcnn/train.log*. Please check the *MaskRCNN/train.py* (this could cost you 1 $\sim$ 2 hours for a good laptop with CPUs).

c)[10 points] Evaluate the Mask RCNN

Finally, we'd like to evaluate the performance of Mask RCNN. So You are required to implement a function which is able to compute the mAP0.5 for both the detection and segmentation tasks. However, the implementation of data preparation may vary by student. We, therefore, provide a separate dataset called *SingleShapeDatast* which provides only one shape in each image. You will evaluate your Mask RCNN with this specific dataset. Please complete the *MaskRCNN/test.py*. Similar results could be found in Fig.5. Note that, due to the limited training steps, the performance would not be perfect. The points would be counted as long as the results are reasonable :)

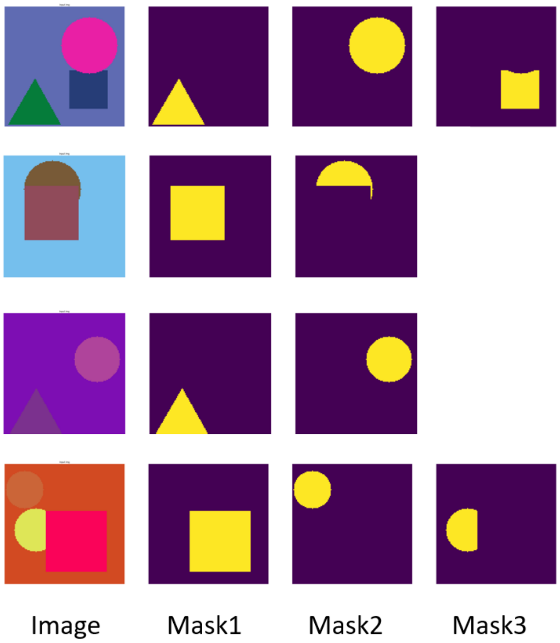

Figure 4: Input data for Mask RCNN

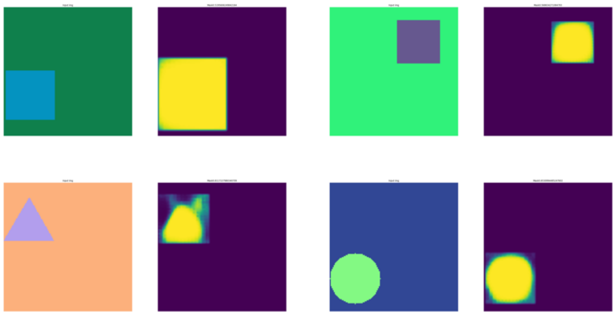

Figure 5: Results of the Mask RCNN