Weekly Progress Report

Jan 03rd - Jan 07th, 2022

Presented by Yannis (Yiming) He 84189287

Noah's Ark | Autonomous Driving Lab
LiDAR Domain Adaptation

Manager: Bingbing Liu 00435285
Supervisor: Eduardo Corral Soto 00407762

HUAWEI

Weekly Summary

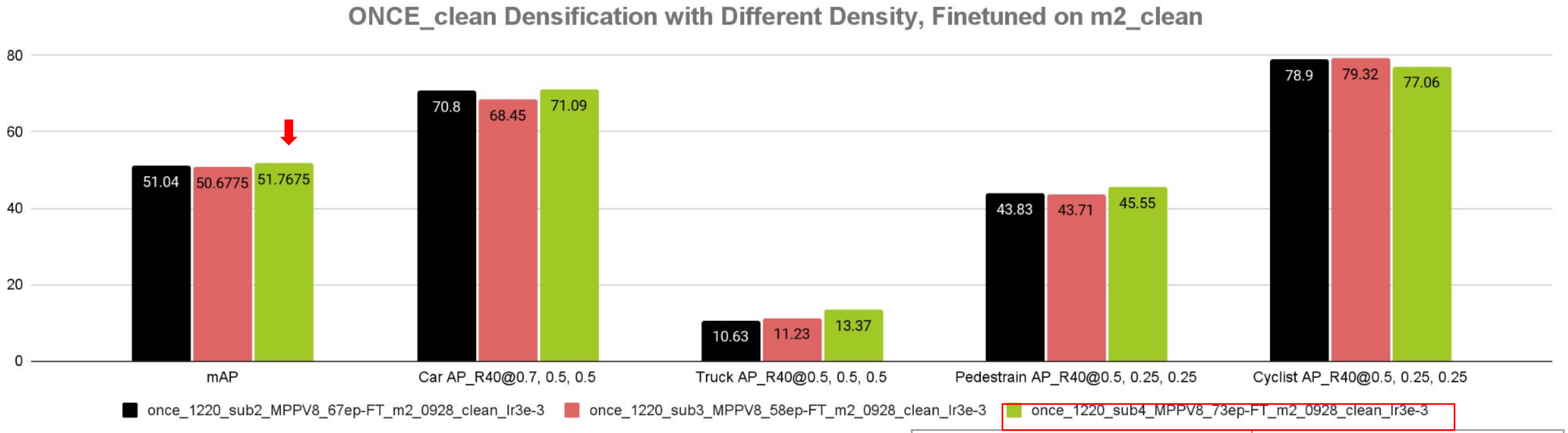
- **Done:**
  - Experiments on clean once,m2 datasets with densified pointcloud (different densification subsampling)
  - MFD parameter experiment (optimize the hyper-parameters)
    - M2 Finetunne_Learning Rate (FTLR)= 3e-5, 3e-4, 3e-3
- **In Progress:**
  - Merging densified once objects with original m2 objects for m2 fine-tuning (avoid forgetting source domain features)
- **TODO**:
  - Incorporate Mrigank's DSBN
  - MFD range experiment
    - No need to densified nearby object. Only densify far-away objects

Work Logs

- Jan 03 (Monday): Holiday
- Jan 04 (Tuesday):
    - Recall Experiment Plans

#	Train	Finetune	Eval
		Baseline	
1,2,3	Cleaned once_1220_sub 2,3,4 with MPPV = 8	Cleaned m2_0928	Cleaned m2_0928
		Parameter Tuning: LR	
4,5,6,7	Cleaned once_1220_sub_best with MPPV = 8	Cleaned m2_0928, lr = 3e-3,4,5,6	Cleaned m2_0928
		Parameter Tuning: ...	

- **Experiments on clean once,m2 datasets with densified pointcloud (different densification subsampling)**



Experiment Variable: Density of the Densification

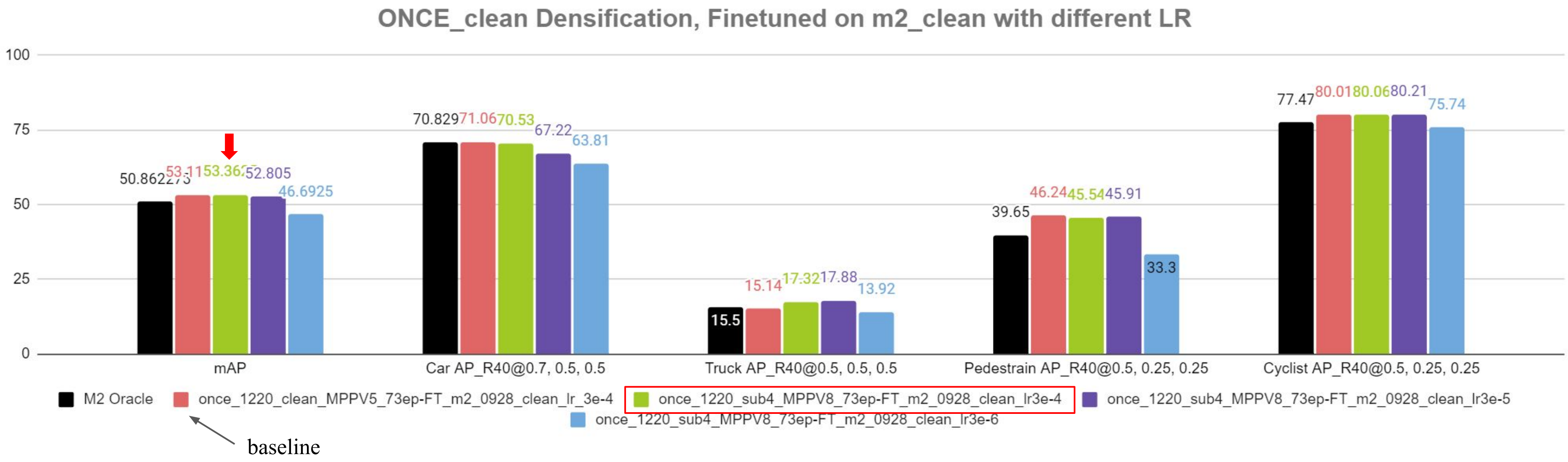
- recall

Name	once:m2 mean#points ratio
m2_0928_clean	1
Once_1220_clean (original)	1.25
once_1220_sub2	0.72
once_1220_sub3	0.83
once_1220_sub4	0.91

Conclusion:

- Sub4 has best performance among all densification
  - Closest density to the target domain (m2)
- Once_1220_sub4 will be used for further experiment

Work Logs:



Experiment Variable:

- LR: 3e-3, 3e-4, 3e-5, 3e-6

Conclusion:

- lr = 3e-4 has best performance
- Once_1220_sub4-FT_m2_lr3e-4 will be used for further experiment

Work Logs

- Jan 04 (Tuesday) (cont'):
    - Add baseline experiment:
        - once_1220_clean_MPPV5_oracle
        - once_1220_clean_MPPV5_73ep-FT_m2_0928_clean_lr_3e-4
- Jan 05-06 (Wed-Thursday)
    - Adding ONCE object to m2_finetuning to avoid forgetting source domain feature
        - Explanation:
            - During the fine-tuning process on target domain, the model starts forgetting source domain
            - → we want to add source samples in data augmentation of the fine-tuning process
                - Inspiration: adding Carla into M2 fine-tuning
            - → i.e. add ONCE samples during fine-tuning on M2
        - Incorporating code base that "Merge Carla & M2 for augmentation" from Aich & Yang
            - Modify the codebase to work with ONCE densified data (in progress)
        - Experiment TODO:
            - Baseline experiment with default data augmentation sample group:
                - ['Car:5', 'Truck:9', 'Pedestrian:6', 'Cyclist:6']
            - Parameter tuning for optimal sample group:
                - Baseline experiment with default data augmentation sample group:
                    - ['Car:?', 'Truck:?', 'Pedestrian:?', 'Cyclist:?']

End of January 07th, Weekly Report

Weekly Progress Report

Jan 10th - Jan 14th, 2022

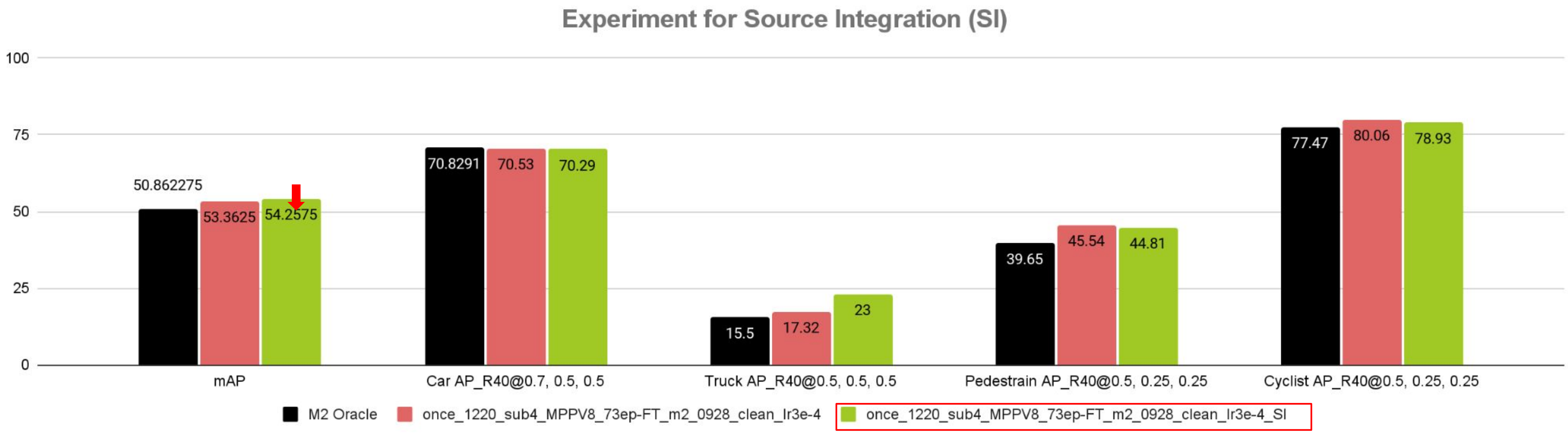
Presented by Yannis (Yiming) He 84189287

Noah’s Ark | Autonomous Driving Lab
LiDAR Domain Adaptation

Manager: Bingbing Liu 00435285
Supervisor: Eduardo Corral Soto 00407762

Weekly Summary

- **Done:**
  - Source Integration:
    - Merging densified once objects with original m2 objects for m2 fine-tuning (avoid forgetting source domain features)
      - Adapt the pipeline from Carla (3 channels) dataset to work with ONCE dataset (4 channels)
    - Parameter tuning: #object insertion: `[‘Car:a’, ‘Truck:b’, ‘Pedestrian:c’, ‘Cyclist:d’]`
- **In Progress:**
  - Add visualization for 1. Before insertion, 2. After insertion, 3. Prediction result   (in progress)
  - Find “good vs bad insertions” by analyzing loss after each insertion
  - Clean “bad” data for other classes (currently, only car point clouds are clean)
- **TODO**:
  - Incorporate Mrigank’s DSBN
  - MFD range experiment
    - No need to densified nearby object. Only densify far-away objects



Experiment Setup:

- Adding source domain (ONCE_MFD_clean) object to the target domain (M2_clean) fine tuning
- avoid forgetting source domain features

Experiment Variable:

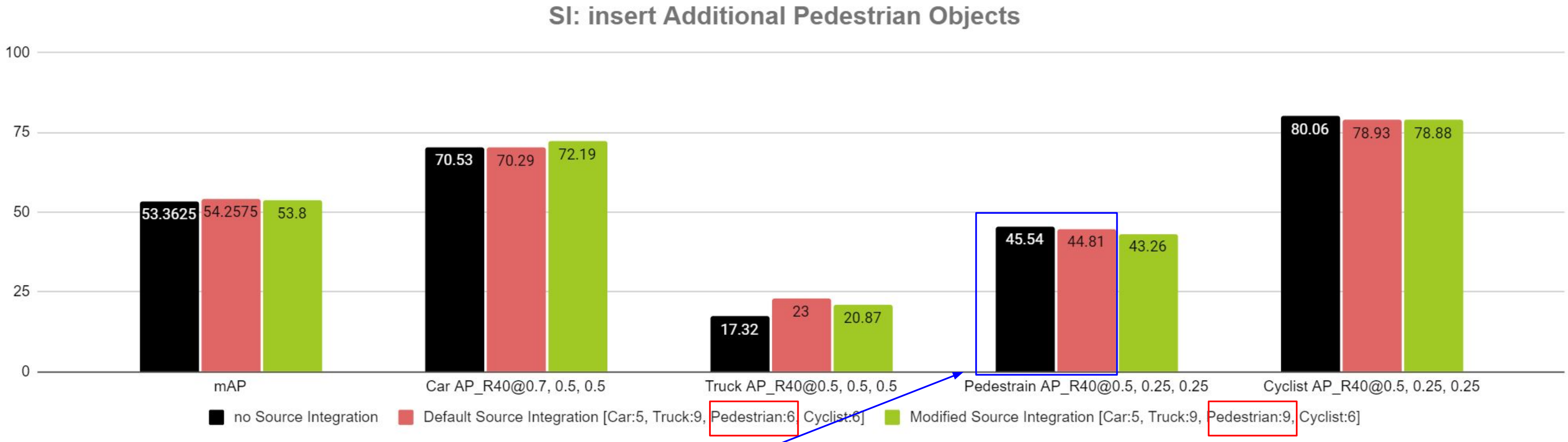
- With vs without source integration
    - Default #object insertion: `['Car:5', 'Truck:9', 'Pedestrian:6', 'Cyclist:6']`

Conclusion:

- Source Integration (SI) improves performance even in default setting
- Major increase in the truck category

Next Step:

- Parameter tuning: #object insertion: `['Car:a', 'Truck:b', 'Pedestrian:c', 'Cyclist:d']`



Experiment Setup:

- Recall: Tuning Hyper-parameters for Source Integration (recall: avoid forgetting source domain features)
- Since we have seen a Pedestrian AP drop, my hypothesis: "*insert more pedestrian object could boost the performance*"

Experiment Variable:

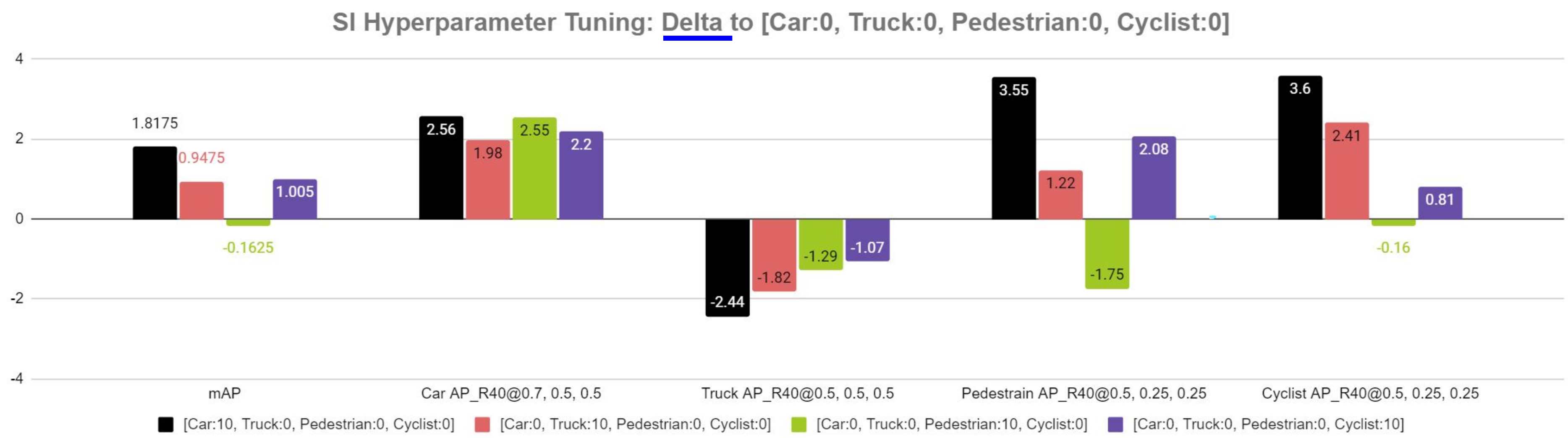
- Parameter tuning: #object insertion: `['Car:a', 'Truck:b', 'Pedestrian:c', 'Cyclist:d']`

Conclusion:

- Reject the hypothesis: "simply *insert more pedestrian would increase the AP of pedestrian*"

Next Step:

- Start learning the reaction to object insertion for each class: `['Car:0', 'Truck:0', 'Pedestrian:0', 'Cyclist:0']`
  - Tuning 1 class at a time



Experiment Setup:

- Baseline: `['Car:0', 'Truck:0', 'Pedestrian:0', 'Cyclist:0']`
- Add 10 to each class individually, i.e.
    - `['Car:10', 'Truck:0', 'Pedestrian:0', 'Cyclist:0']`
    - …
    - `['Car:0', 'Truck:0', 'Pedestrian:0', 'Cyclist:1 0']`

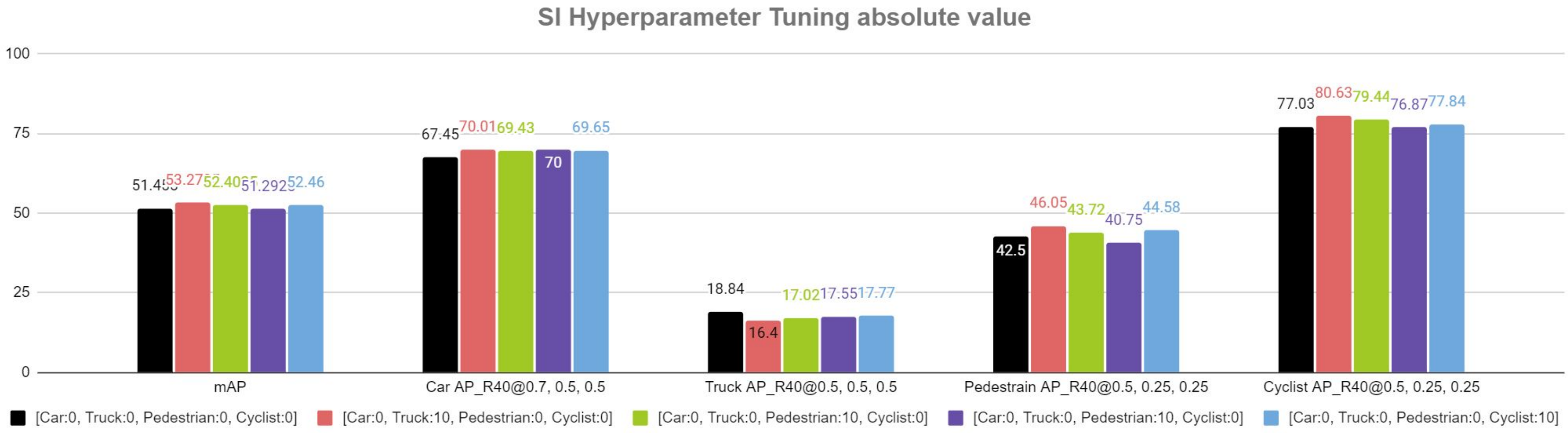
Conclusion:

- Increasing “car” class has the best return over all classes → since the dataset clean the “bad” car pointcloud
- Truck class has negative response to any classes’ insertion

Next Step:

- Add visualization for 1. Before insertion, 2. After insertion, 3. Prediction result (in progress)
- Find “good vs bad insertions” by analyzing loss after each insertion
- Clean “bad” data for other classes (currently, only car point clouds are clean)

Appendix:



SI Hyperparameter Tuning absolute value
100
75
50
25
0
51.456
53.275
52.405
51.292
52.46
67.45
70.01
69.43
70
69.65
18.84
16.4
17.02
17.55
17.77
42.5
46.05
43.72
40.75
44.58
77.03
80.63
79.44
76.87
77.84
mAP
Car AP_R40@0.7, 0.5, 0.5
Truck AP_R40@0.5, 0.5, 0.5
Pedestrain AP_R40@0.5, 0.25, 0.25
Cyclist AP_R40@0.5, 0.25, 0.25
[Car:0, Truck:0, Pedestrian:0, Cyclist:0]
[Car:0, Truck:10, Pedestrian:0, Cyclist:0]
[Car:0, Truck:0, Pedestrian:10, Cyclist:0]
[Car:0, Truck:0, Pedestrian:10, Cyclist:0]
[Car:0, Truck:0, Pedestrian:0, Cyclist:10]

End of January 14th, Weekly Report