POINT GNN

CS6450:VISUAL COMPUTING

Presented by- Kaushiki Dwivedi (M.Tech ,Artificial Intelligence -AI21MTECH14003)

AUTHORS-
Shi,Weijing
Rajkumar,Ragunathan (Raj)

TEACHING ASSISTANT:
Romi Srivastava
Ph.D Research Scholar

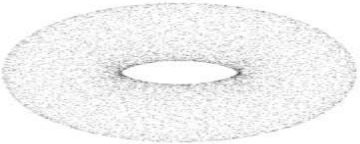
GUIDED BY:
Prof. C Krishna Mohan
Dept. of CSE, IIT Hyderabad

"Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud"

Weijing Shi and Ragunathan (Raj) Rajkumar Carnegie Mellon University Pittsburgh, PA 15213 {weijings, rajkumar}@cmu.edu "

The IEEE Conference on Computer Vision and Pattern Recognition **(CVPR),2020**

Table of Contents:



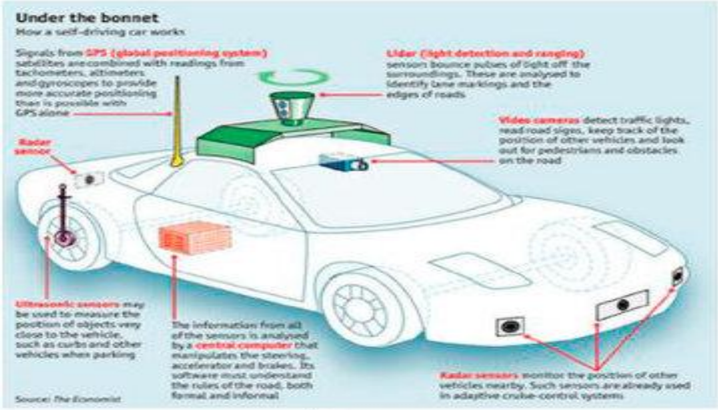
RECAP: POINT CLOUD

MOTIVATION:

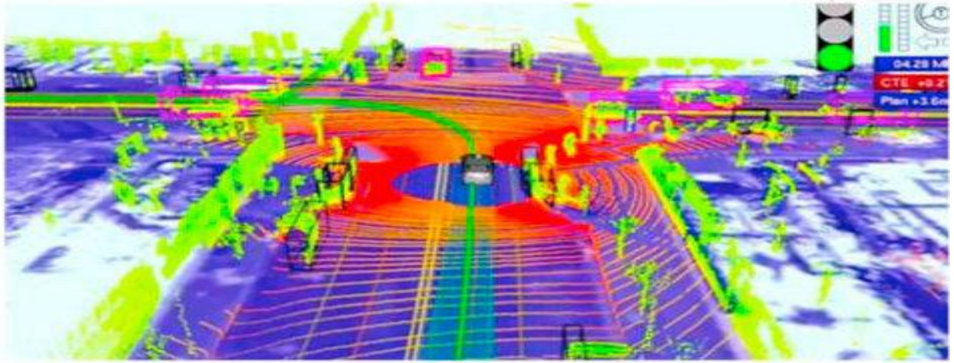
- Application areas:
  - Self driving cars.
  - 3D CAD models
  - Industrial Metrology
  - Quality inspection
  - Speech recognition
- New approach for 3D object detection

CHALLENGES

- A high-density LiDAR usually leads to a high cost,expensive tech.
- Robustness on LiDAR sparsity.
- Enormous amount of data ,few seconds of data even on a low level 32 channel gets into GB of data.
- Scenarios like fog,rain etc that blocks light can affect the working of LiDAR.



a)



b)

GRAPH NEURAL NETWORKS

The feature vector of the node v

The edge feature vector of the edge (v, u)

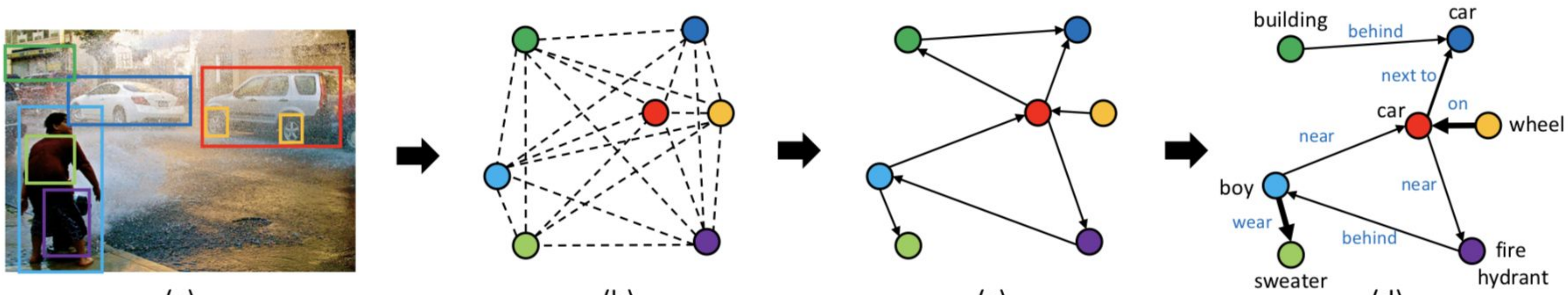
The feature vector for the neighboring node u

$$\mathbf{h}_v^{(t)} = \sum_{u \in N(v)} f(\mathbf{x}_v, \mathbf{x^e}_{(v,u)}, \mathbf{x}_u, \mathbf{h}_u^{(t-1)})$$

The hidden feature vector of node v at time t

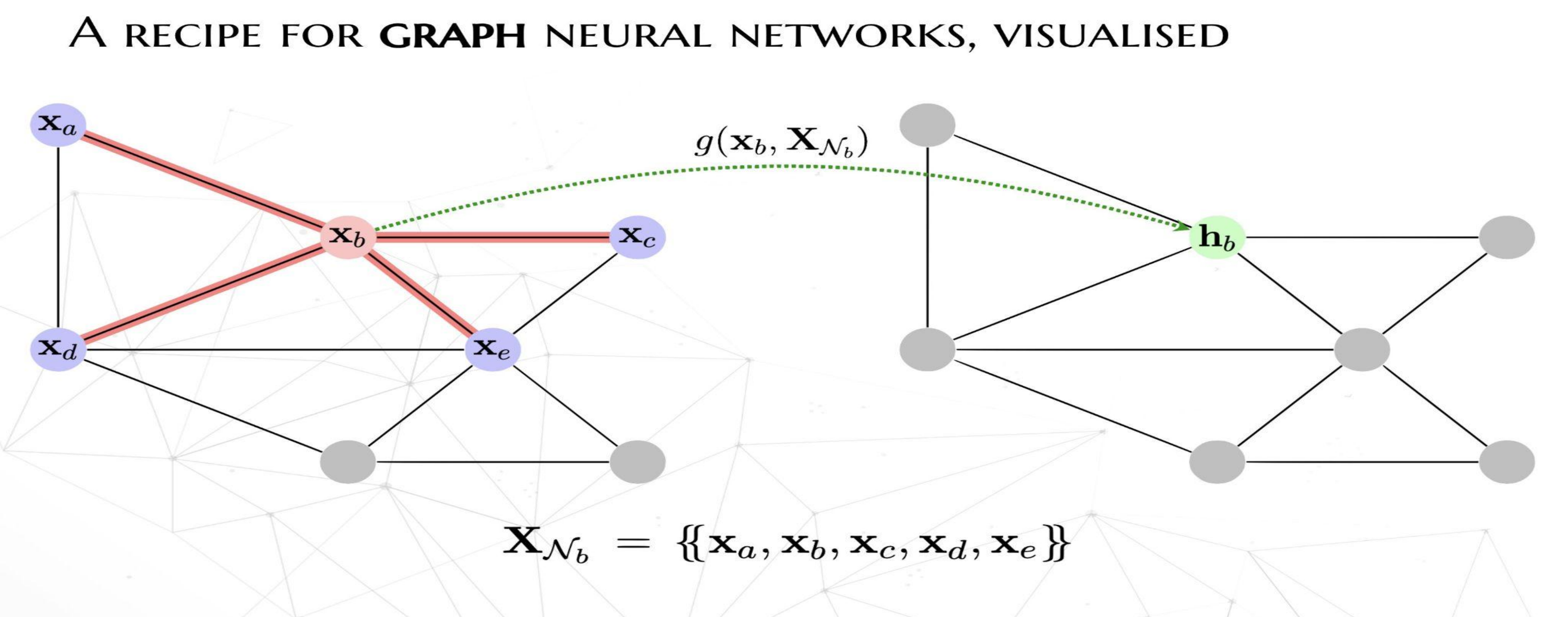
Neighbors of node v

The hidden feature vector of node u in last time step



(a) (b) (c) (d)

VISUALIZING GNN:



Contributions of the paper:

- Propose a new object detection approach using graph neural network on the point cloud.
- The paper designs Point-GNN, a graph neural network with an auto-registration mechanism that detects multiple objects in a single shot.
- Able to achieve state-of-the-art 3D object detection accuracy in the KITTI benchmark and analyze the effectiveness of each component in depth.

MODEL ARCHITECTURE:

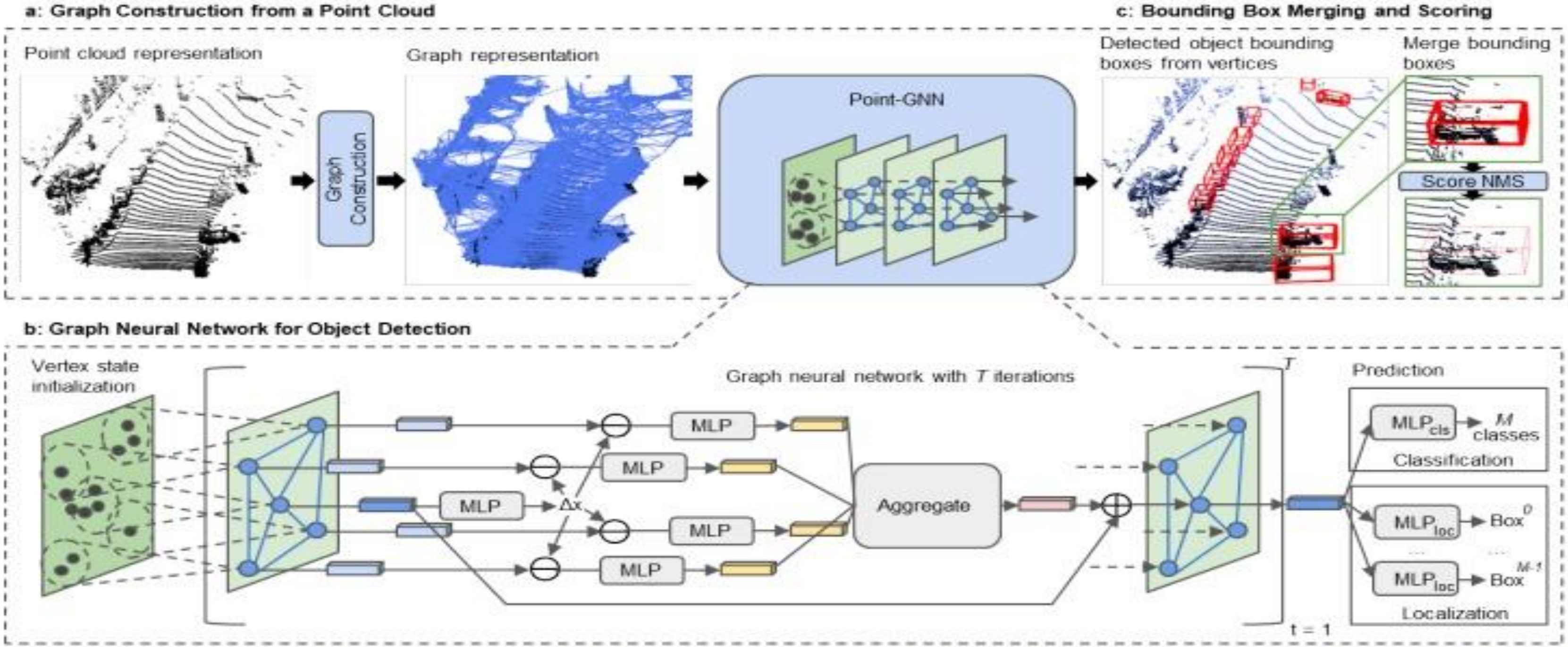
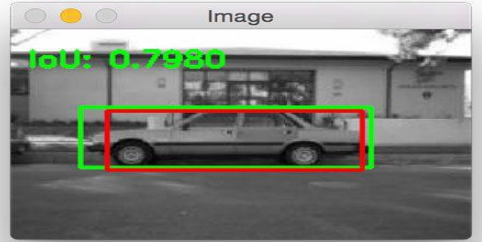
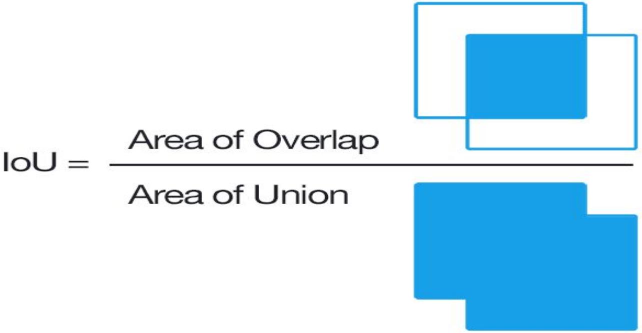
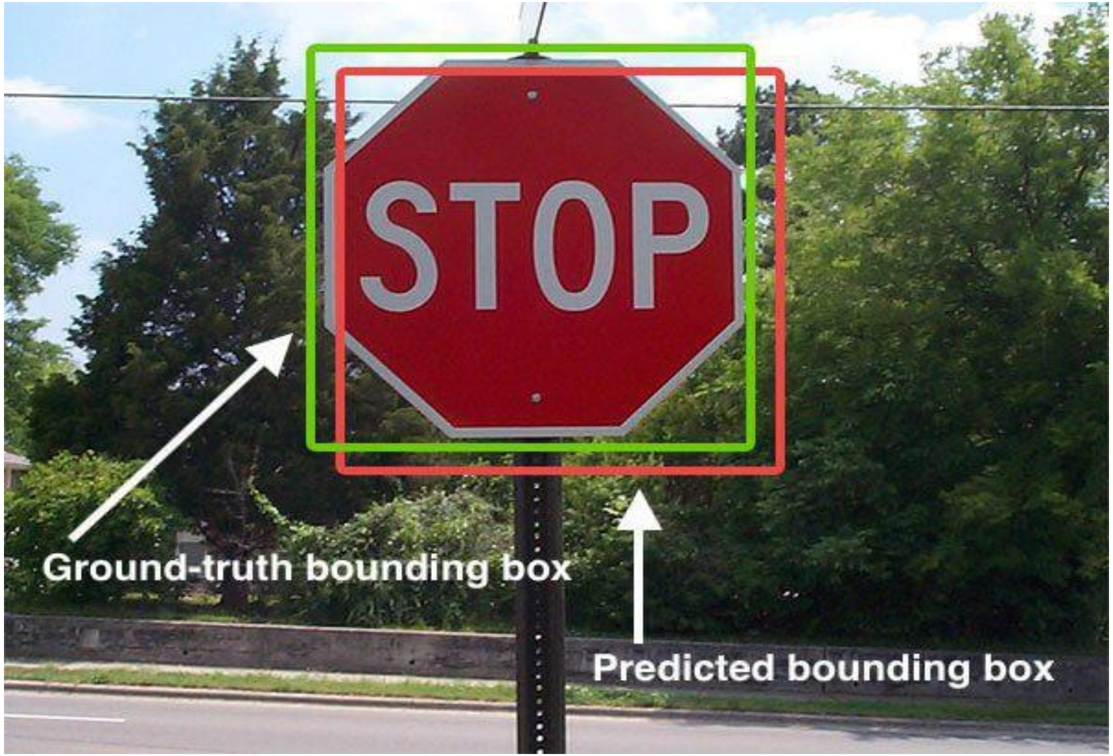


Figure 2. The architecture of the proposed approach. It has three main components: (a) graph construction from a point cloud, (b) a graph neural network for object detection, and (c) bounding box merging and scoring.

INTERSECTION OF UNION:EVALUATION METRIC







NMS ALGORITHM & MODIFIED USING BOX MERGING AND SCORING

$\mathcal{M} \leftarrow \{\}, \mathcal{Z} \leftarrow \{\}$
while $\mathcal{B} \neq empty$ **do**
  $i \leftarrow argmax\ D$
  $\mathcal{L} \leftarrow \{\}$
  for $b_j\ in\ \mathcal{B}$ **do**
    if $iou(b_i, b_j) > T_h$ **then**
      $\mathcal{L} \leftarrow \mathcal{L} \cup b_j$
      $\mathcal{B} \leftarrow \mathcal{B} - b_j, \mathcal{D} \leftarrow \mathcal{D} - d_j$
    end
  end
  $m \leftarrow median(\mathcal{L})$
  $o \leftarrow occlusion(m)$
  $z \leftarrow (o+1) \sum_{b_k \in \mathcal{L}} IoU(m, b_k) d_k$
  $\mathcal{M} \leftarrow \mathcal{M} \cup m, \mathcal{Z} \leftarrow \mathcal{Z} \cup z$
end
return $\mathcal{M}, \mathcal{Z}$

Loss Function:

- **Classification loss:**

$$l_{cls} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{c_j}^{i} \, log(p_{c_j}^{i})$$

- **Localisation loss:**

$$l_{loc} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(v_i \in b_{interest}) \sum_{\delta \in \delta_{b_i}} l_{huber}(\delta - \delta^{gt})$$

DATASETS:

- The KITTI dataset contains 7481 training samples and 7518 testing samples.
- Train the proposed GNN end-to-end with a batch size of 4.
- Each sample provides both the point cloud and the camera image.
- The KITTI benchmark evaluates the **average precision** (AP) of three types of objects:

  Car, Pedestrian and Cyclist.

- **Conda environments: fusion_env_3.6,keras_env,open3d_39,pytorch_p37**
- **GTX 1070 GPU and Xeon E5-1630 CPU.**

Implementation:

Used three iterations **(T = 3)** in our GNN. During training, we limit the maximum number of input edges per vertex to **256.**

- **Car:-**

  -Treat front view and side-view objects as two different classes.

  -We use an initial learning rate of 0.125 and a decay rate of 0.1 every 400K steps.

  -Trained it for 1400K steps.

- **Pedestrian and Cyclist:**

  -We use an initial learning rate of 0.32 and a decay rate of 0.25 every 400K steps.

  - Trained it for 1000K steps.

POINT-GNN CODE: run.py

Extract Parsed Arguments

DATASET

Prepare Sequence data for training and validation splits of dataset

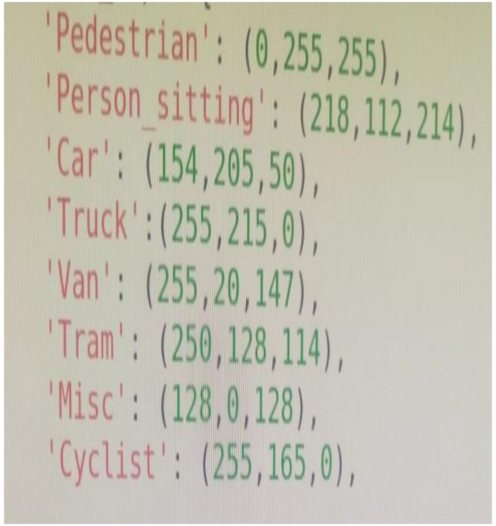
Dataloader class

Preparing dataset for class:CAR,PEDESTRIAN and CYCLIST by dividing in batches

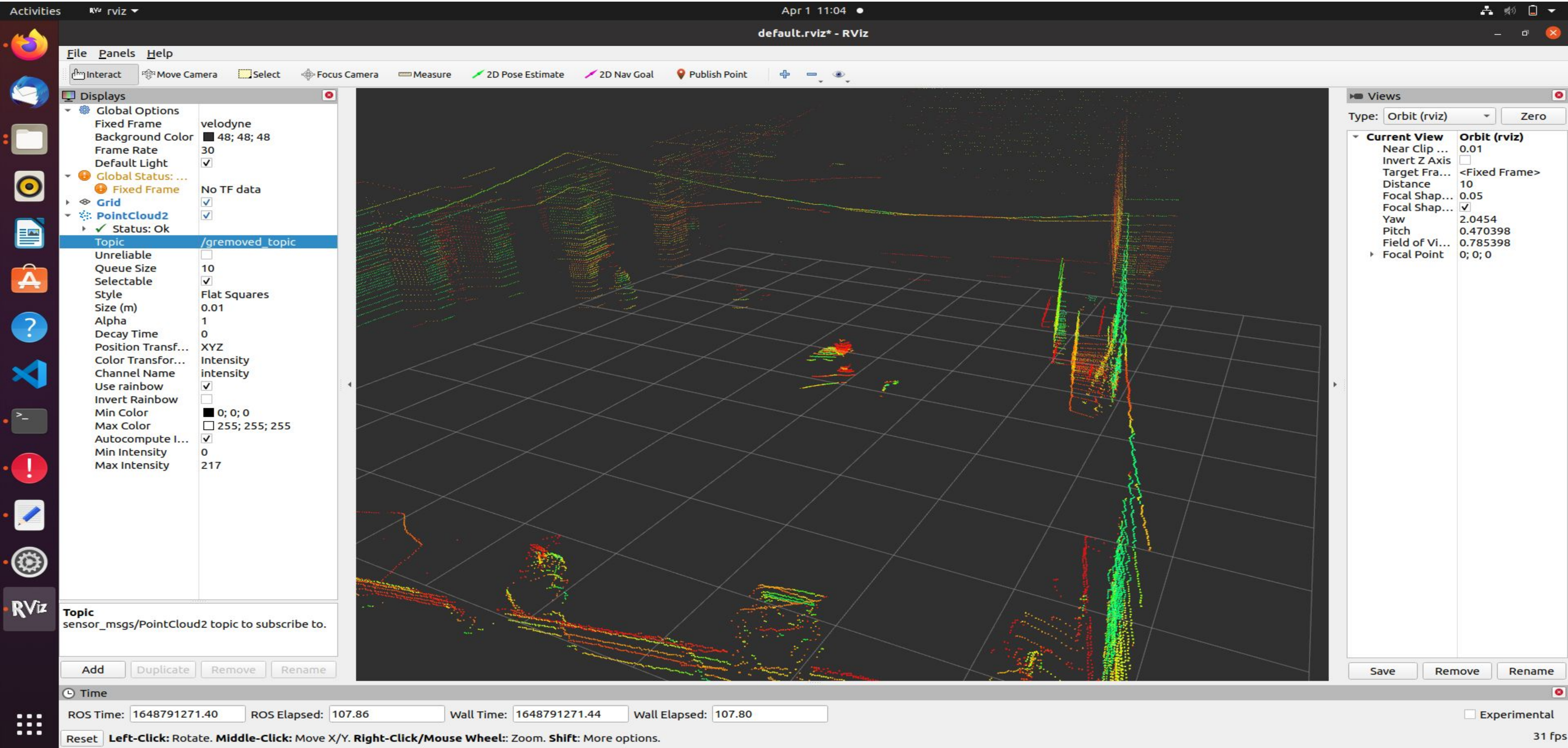
Batches:
(Front view and side view of car)
(2nd n/w into pedestrian and cyclist)

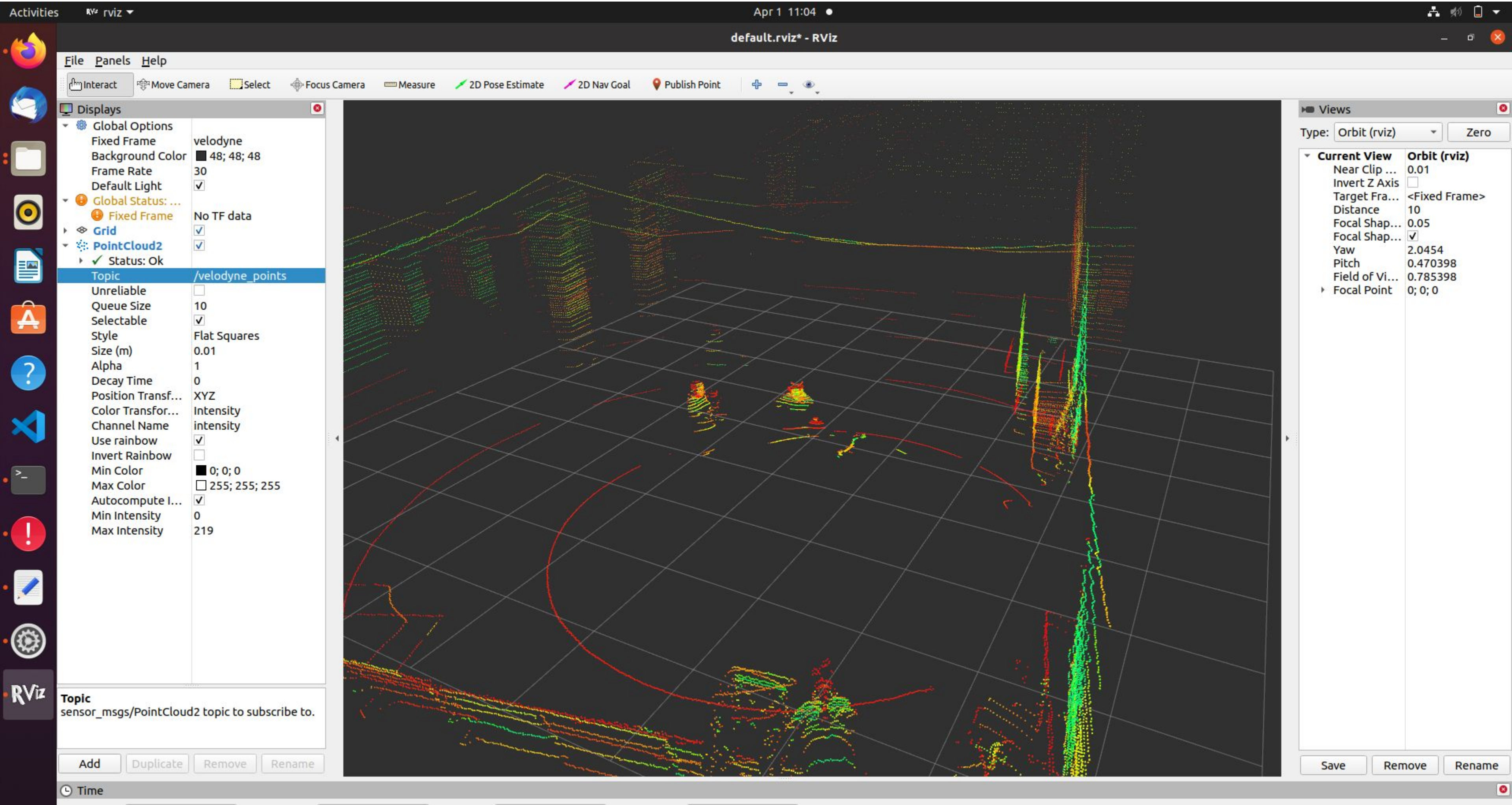
Load Pretrained model and compute avg loss

Stored at Checkpoint folder

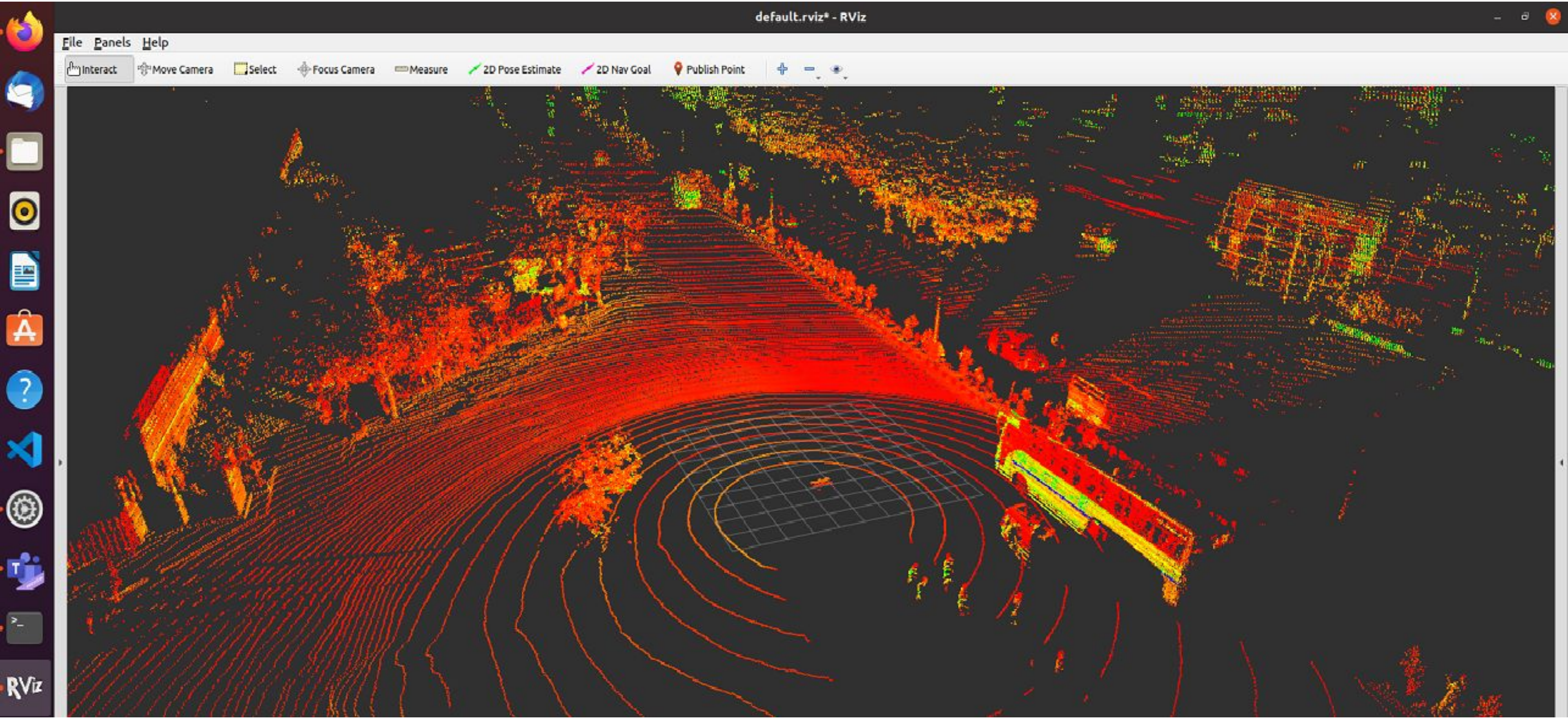


Results obtained by paper implementation:



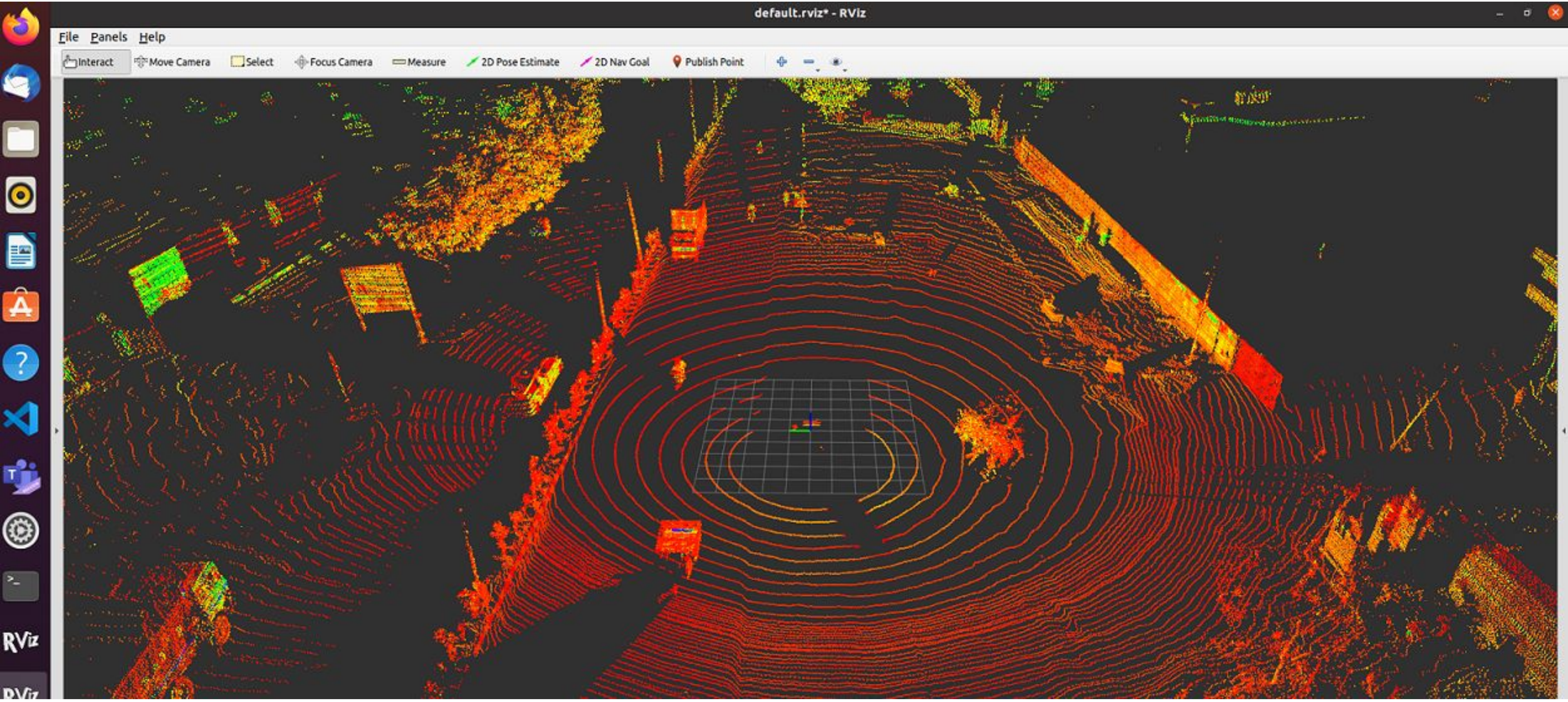

Activities
rviz
Apr 1 11:04
default.rviz* - RViz
File
Panels
Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Fixed Frame
velodyne
Background Color
48; 48; 48
Frame Rate
30
Default Light
Global Status: ...
Fixed Frame
No TF data
Grid
PointCloud2
Status: Ok
Topic
/velodyne_points
Unreliable
Queue Size
10
Selectable
Style
Flat Squares
Size (m)
0.01
Alpha
1
Decay Time
0
Position Transf...
XYZ
Color Transfor...
Intensity
Channel Name
intensity
Use rainbow
Invert Rainbow
Min Color
0; 0; 0
Max Color
255; 255; 255
Autocompute I...
Min Intensity
0
Max Intensity
219
Topic
sensor_msgs/PointCloud2 topic to subscribe to.
Add
Duplicate
Remove
Rename
Views
Type:
Orbit (rviz)
Zero
Current View
Orbit (rviz)
Near Clip ...
0.01
Invert Z Axis
Target Fra...
<Fixed Frame>
Distance
10
Focal Shap...
0.05
Focal Shap...
Yaw
2.0454
Pitch
0.470398
Field of Vi...
0.785398
Focal Point
0; 0; 0
Save
Remove
Rename
Time

Results:



default.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
RViz

Results(Contd):

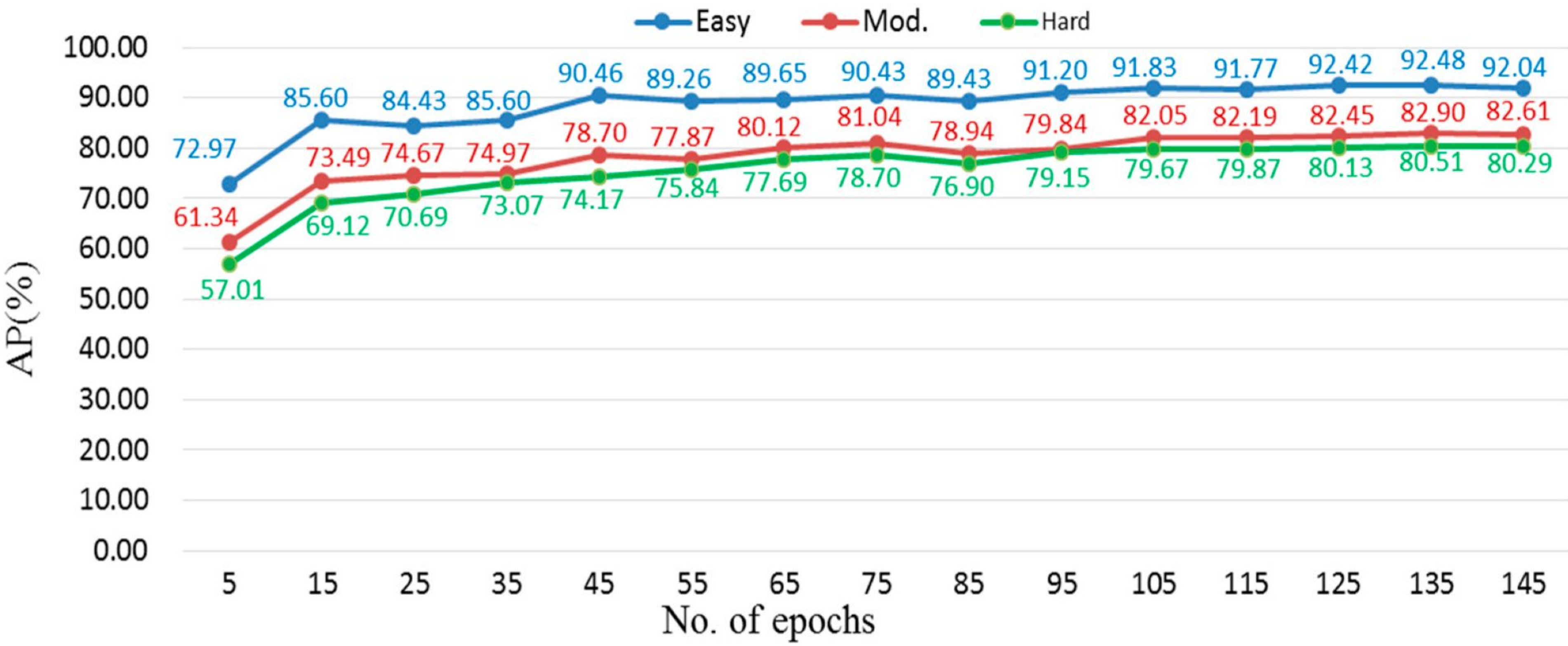


Results:

- **3D object detection on the KITTI test dataset.**

Method	Modality	Car			Pedestrian			Cyclist		
		Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
UberATG-ContFuse[12]	LiDAR + Image	82.54	66.22	64.04	N/A	N/A	N/A	N/A	N/A	N/A
AVOD-FPN[8]	LiDAR + Image	81.94	71.88	66.38	50.80	42.81	40.88	64.00	52.18	46.61
F-PointNet[13]	LiDAR + Image	81.20	70.39	62.19	51.21	44.89	40.23	71.96	56.77	50.39
UberATG-MMF[11]	LiDAR + Image	86.81	76.75	68.41	N/A	N/A	N/A	N/A	N/A	N/A
VoxelNet[23]	LiDAR	81.97	65.46	62.85	**57.86**	**53.42**	**48.87**	67.17	47.65	45.11
SECOND[19]	LiDAR	83.13	73.66	66.20	51.07	42.56	37.29	70.51	53.85	53.85
PointPillars[10]	LiDAR	79.05	74.99	68.30	52.08	43.53	41.49	75.78	59.07	52.92
PointRCNN[16]	LiDAR	85.94	75.76	68.32	49.43	41.78	38.63	73.93	59.60	53.59
STD[21]	LiDAR	86.61	77.63	**76.06**	53.08	44.24	41.97	**78.89**	62.53	55.77
Our Point-GNN	LiDAR	**88.33**	**79.47**	72.29	51.92	43.77	40.14	78.60	**63.48**	**57.08**

Illustration:



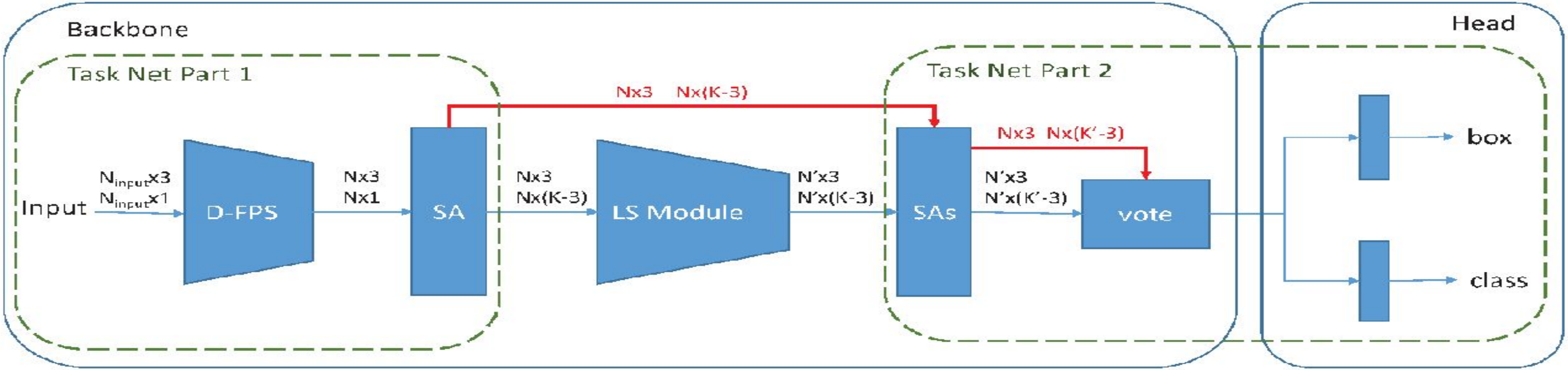
Ablation study Results:

Comparing the Object detection accuracy with and without box merging and scoring:

	Box Merge	Box Score	Auto Reg.	BEV AP (Car) Easy	BEV AP (Car) Moderate	BEV AP (Car) Hard	3D AP (Car) Easy	3D AP (Car) Moderate	3D AP (Car) Hard
1	-	-	-	89.11	87.14	86.18	85.46	76.80	74.89
2	-	-	✓	89.03	87.43	86.39	85.58	76.98	75.69
3	✓	-	✓	89.33	87.83	86.63	86.59	77.49	76.35
4	-	✓	✓	89.60	88.02	86.97	87.40	77.90	76.75
5	✓	✓	-	90.03	88.27	87.12	88.16	78.40	77.49
6	✓	✓	✓	89.82	88.31	87.16	87.89	78.34	77.38

Novel Work Contribution:

- Propose the learned sampling network (LSNet), a single-stage 3D object detection network containing an LS module that can sample important points through deep learning.
- A novel deep learning based sampling approach i.e. differentiable and task related.



Conclusion:

- ❖ Using a graph representation, encoding the point cloud compactly without mapping to a grid and grouping repeatedly.
- ❖ Point-GNN achieves the leading accuracy in both the 3D and Bird’s Eye View object detection of the KITTI benchmark.
- ❖ Experiments show the proposed auto-registration mechanism reduces transition variance, and the box merging and scoring operation improves the detection accuracy.

References

- B. Yang, W. Luo, and R. Urtasun. Pixor: Real-time 3d object detection from point clouds. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7652– 7660, June 2018.
- Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. Multi-task, multi-sensor fusion for 3d object detection. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.
- X. Qi, R. Liao, J. Jia, S. Fidler, and R. Urtasun. 3d graph neural networks for rgbd semantic segmentation. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 5209–5218, Oct 2017.
- J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander. Joint 3d proposal generation and object detection from view aggregation. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 1–8, Oct 2018.
- Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. Sensors, 18(10), 2018.

THANKYOU