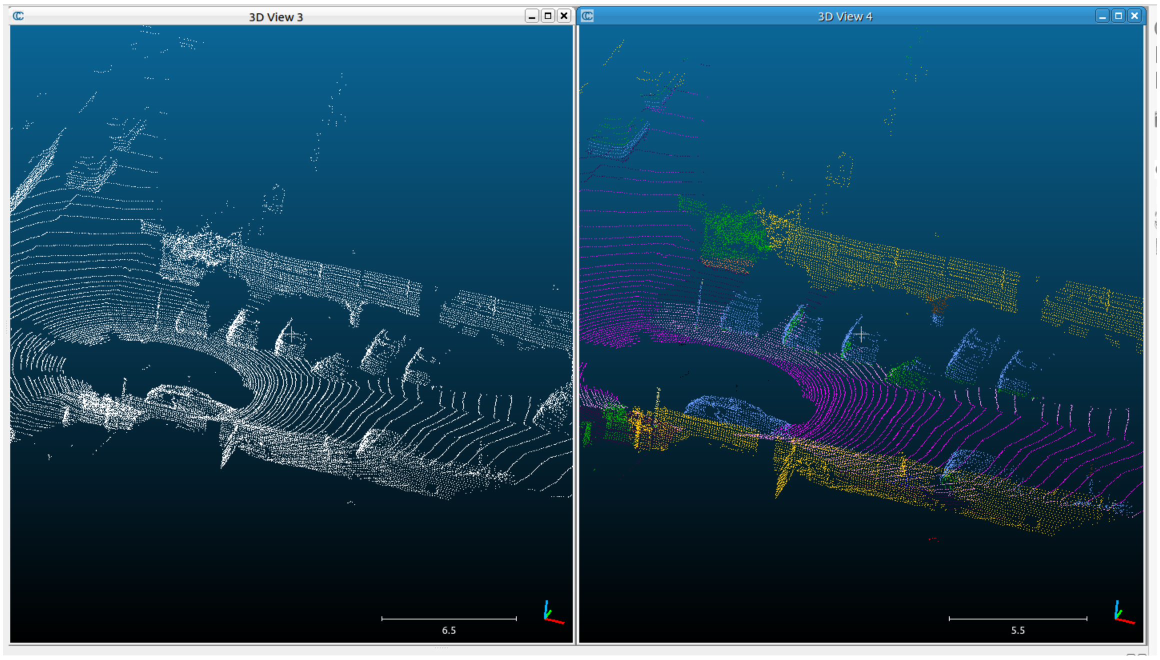
PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation



A. S. Cédric KUASSIVI
Engineer in Digital Sciences
+33 (0)6 18 03 50 86
cedrickuassivi@gmail.com

Contents

List of Figures

1 Context and Issues

For this project, I decided to implement the pointnet network from scratch and train the model for semantic segmentation task on the semantic-kitti dataset. The source code of the project is available here: https://github.com/KASCedric/PointNet.

1.1 PointNet neural network

PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation [3] is a research paper published in 2017. It deals with a neural network: pointnet allowing the part/semantic segmentation and classification of 3D point clouds (see figure 1).

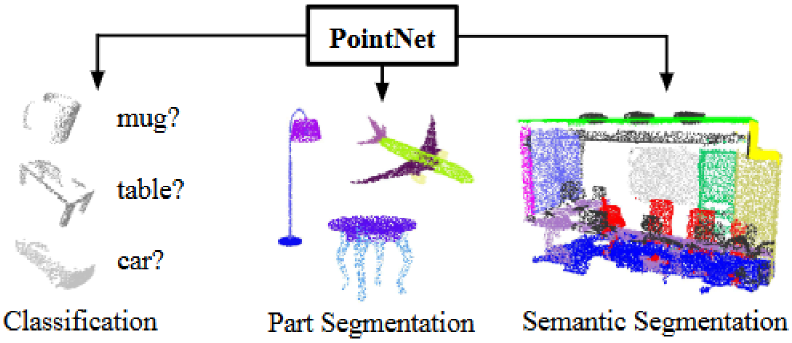


Figure 1 – Applications of PointNet

Unlike classical neural networks applied to point clouds, which take voxel grids or collections of images as input, the strength of pointnet neural network is its capacity to take a raw point cloud as input. Thus, it avoids unnecessarily voluminous data and quantization artifacts that can obscure natural invariances of the data.

PointNet neural network (see figure 2) takes as input point clouds that have the following properties:

- **Unordered**: This implies that regardless of the order of the points of a cloud, it remains the same. The use of a max pooling layer as a symmetric function allows this property to be considered.
- **Interaction among points**: The points and their neighborhood are used to describe the content of a point cloud. Thus, a structure for extracting local and global features is used in the neural network.
- **Invariance under transformations**: A rotation or translation of a point cloud does not change its nature. A joint alignment network (TNet) is integrated into the pointnet architecture to make it robust to rigid transformations.

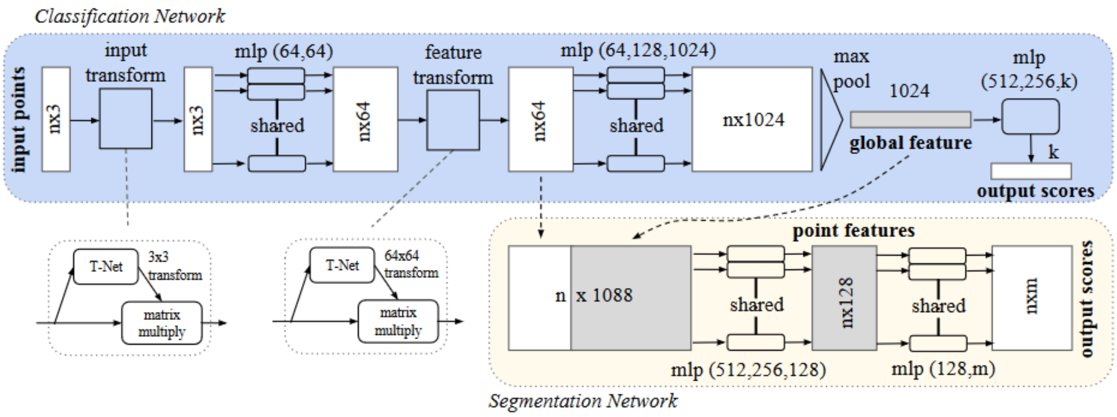


Figure 2 – PointNet Architecture

1.2 Semantic-Kitti dataset

The kitti odometry dataset [2] is a set of data containing 3D points from cityscape scenes. This dataset is composed of 22 sequences (from 00 to 21), each containing a certain number of point clouds.

The semantic-kitti dataset [1] offers point by point annotations (see figure 3) of the sequences 00 to 10 of the kitti odometry dataset. These data will allow supervised training of neural networks in the semantic segmentation of point clouds. I trained the pointnet network on the first sequence (sequence 00) of the semantic-kitti dataset, which contains 4540 point clouds of about 100,000 points each.

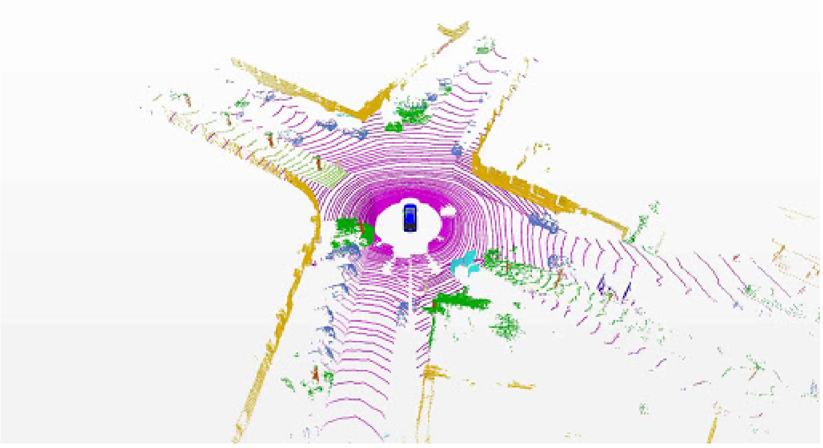


Figure 3 – Single scan of semantic-kitti dataset

In this report, I will first describe the materials and methods used to achieve my objectives. I will finish by presenting the results obtained and discuss their relevance.

2 Materials and Methods

2.1 Resources

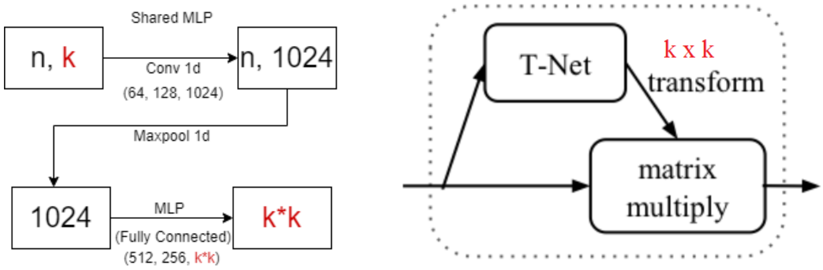
- I trained the model on sequence 00 of semantic-kitti dataset for 17 hours with a GTX 1060 GPU.
- C++: I used the Point Cloud Library (PCL) to handle and process the 3D data, and used Open Multi-Processing framework (OpenMP) to optimize the data processing.
- Python: I used Pytorch machine learning framework to implement, train, and evaluate the semantic segmentation network.

2.2 Progress and Difficulties

This project was carried out in 3 steps.

Step 1: Network implementation
I implemented the semantic segmentation neural network. The difficulty of this step was to understand the usefulness of the layers that the authors of the paper used.

- TNet (see figure 4a) is a joint alignment network used in pointnet architecture. It outputs a square matrix of the same size as its input features. The transform matrix is then applied to the input data of the TNet network (see figure 4b).



(a) TNetkd Architecture

(b) TNet transform matrix

Figure 4 – TNet joint alignment network

A regularization term: $\left\|I - AA^T\right\|_F^2$ (where A is the transform matrix at the output of the TNet) has been added to the loss function to force the transform matrix to be orthogonal. This has the advantage of keeping the information of the input of the TNet network after applying the transform matrix to it.

- One-dimension convolutional layers were used to implement the shared layers:
- The LogSoftmax activation was used for the last layer, instead of the softmax activation, for computational reasons. The negative log likelihood loss (NLLLoss) is the loss associated to the LogSoftmax activation.

To make sure the network was implemented correctly, I generated random tensors and checked that the output tensors had the right size.

Step 2: Data acquisition & processing
The semantic-kitti dataset provides point-by-point annotation of the point clouds in the kitti odometry velodyne 3D dataset (see section 1.2 for more details). I kept all the semantic-kitti classes for the segmentation task.
Most likely due to a bad internet connection I had trouble downloading the 80 GB of kitti odometry velodyne data. I finally managed to download the data using the command line program *wget*.

A python script was written to read the 3D data and labels and then generated a .ply file that I displayed using the cloud compare visualization application. This allowed me to understand the structure of the data: number of points per cloud, number of classes, etc.

I wrote a dataloader to feed the neural network. Given the density of the point clouds (about 100k points per cloud), the vram memory was insufficient. So, I decided to sub-sample the points before training the neural network.

C++ language was used to read and sub-sample point clouds and labels. At the end of the process, the point clouds and their labels were saved in .ply files.
The labels of the points were remapped to correspond to a sequence of numbers from 0 to 33 for the 34 classes of the semantic-kitti dataset. This was done to avoid errors when applying the logsoftmax activation and using the negative log likelihood loss.
Also, I decided to keep the coordinates of the points and their labels in the same file to optimize the dataloader which will open a single file for the features and labels instead of 2 files.

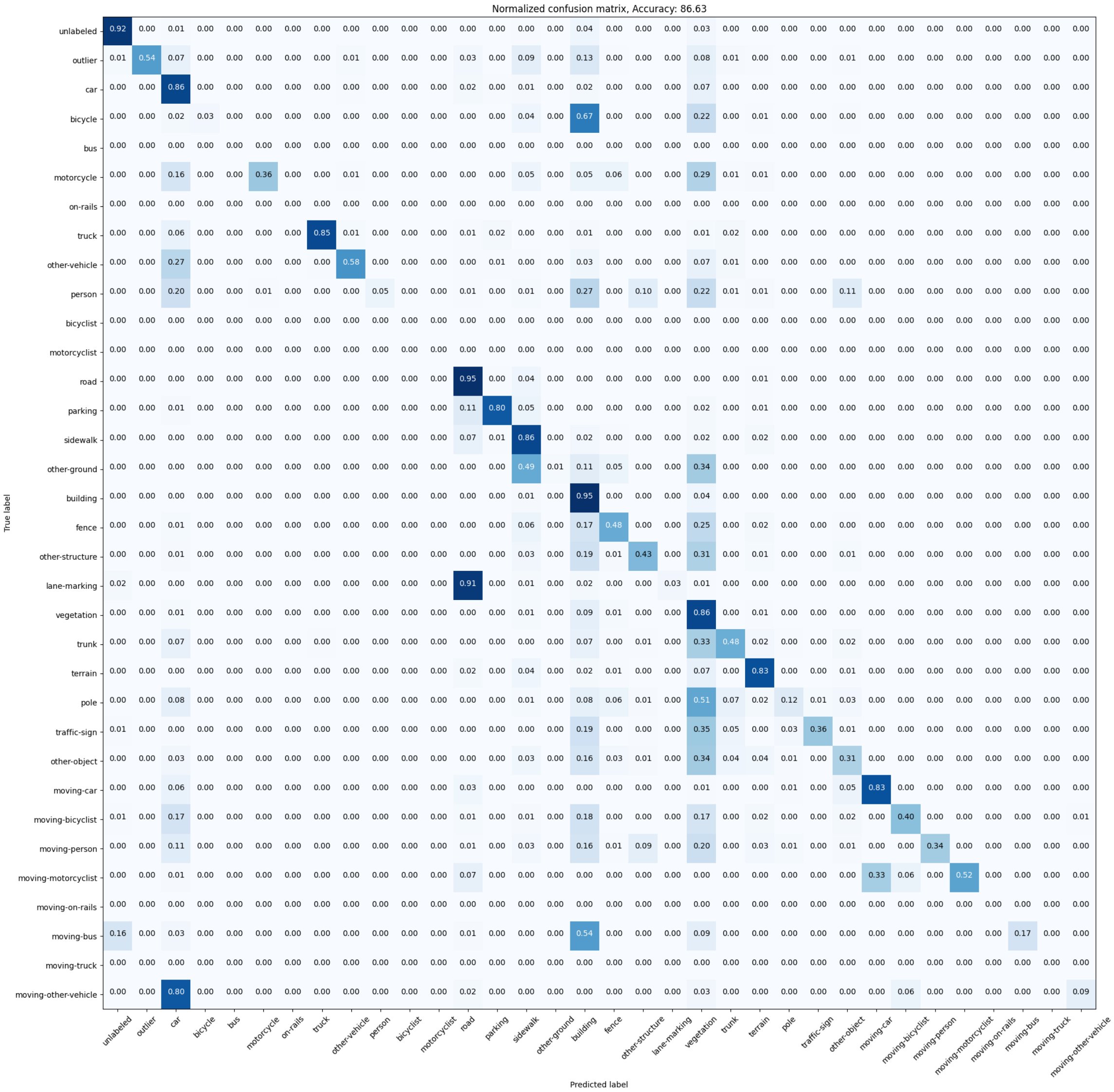
Using a $leaf\_size = 0.1$ (10 cm) for sub-sampling proved sufficient to load the point clouds into vram memory, with a $batch\_size = 1$.
I realized that the batch size would always be 1, again for memory reasons, unless the clouds were significantly sub-sample, which would result in a considerable loss of information.
The other reason was that the point clouds have different numbers of points which causes a concatenation error when creating the batches. One solution would have been to create batches of the same size, but this would have made the code more complex. So, I decided to keep a $batch\_size = 1$ for the training.

Step 3: Training
I only used the sequence 00 of kitti odometry dataset. I divided the 4540 clouds into train, validation and test sets (respectively 3633, 2. 905 point clouds).
I decided to take 2 clouds for validation during training to get an approximate idea of the evolution of the training (underfitting? overfitting?) without increasing considerably the duration of the training.

During the training, I printed out the evolution of the train loss, accuracy and validation loss. The losses decreased while the accuracy increased. This allowed me to ensure that the algorithm converged well. However, I did not do jittering or rotation as mentioned in the pointnet paper.

3 Results and Discussion

After 17 hours of training, I got a model trained on 20 epochs. The accuracy of the model on the test data is 86.63%.



	unlabeled	outlier	car	bicycle	bus	motorcycle	on-rails	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	other-structure	lane-marking	vegetation	trunk	terrain	pole	traffic-sign	other-object	moving-car	moving-bicyclist	moving-person	moving-motorcyclist	moving-on-rails	moving-bus	moving-truck	moving-other-vehicle
unlabeled	0.92	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.04	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
outlier	0.01	0.54	0.07	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.03	0.00	0.09	0.00	0.13	0.00	0.00	0.00	0.08	0.01	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
car	0.00	0.00	0.86	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.01	0.00	0.02	0.00	0.00	0.00	0.07	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
bicycle	0.00	0.00	0.02	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.04	0.00	0.67	0.00	0.00	0.00	0.22	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
bus	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
motorcycle	0.00	0.00	0.16	0.00	0.00	0.36	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.05	0.06	0.00	0.00	0.29	0.01	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
on-rails	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
truck	0.00	0.00	0.06	0.00	0.00	0.00	0.00	0.85	0.01	0.00	0.00	0.00	0.01	0.02	0.00	0.00	0.01	0.00	0.00	0.00	0.01	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
other-vehicle	0.00	0.00	0.27	0.00	0.00	0.00	0.00	0.00	0.58	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.03	0.00	0.00	0.00	0.07	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
person	0.00	0.00	0.20	0.00	0.00	0.01	0.00	0.00	0.00	0.05	0.00	0.00	0.01	0.00	0.01	0.00	0.27	0.00	0.10	0.00	0.22	0.01	0.01	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
bicyclist	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
motorcyclist	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
road	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.95	0.00	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
parking	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.11	0.80	0.05	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
sidewalk	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.07	0.01	0.86	0.00	0.02	0.00	0.00	0.00	0.02	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
other-ground	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.49	0.01	0.11	0.05	0.00	0.00	0.34	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
building	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.95	0.00	0.00	0.00	0.04	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
fence	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.06	0.00	0.17	0.48	0.00	0.00	0.25	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
other-structure	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.19	0.01	0.43	0.00	0.31	0.00	0.01	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
lane-marking	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.91	0.00	0.01	0.00	0.02	0.00	0.00	0.03	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
vegetation	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.09	0.01	0.00	0.00	0.86	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
trunk	0.00	0.00	0.07	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.07	0.00	0.01	0.00	0.33	0.48	0.02	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
terrain	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.04	0.00	0.02	0.01	0.00	0.00	0.07	0.00	0.83	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
pole	0.00	0.00	0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.08	0.06	0.01	0.00	0.51	0.07	0.02	0.12	0.01	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
traffic-sign	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.19	0.00	0.00	0.00	0.35	0.05	0.00	0.03	0.36	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
other-object	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.16	0.03	0.01	0.00	0.34	0.04	0.04	0.01	0.00	0.31	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
moving-car	0.00	0.00	0.06	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.01	0.00	0.05	0.83	0.00	0.00	0.00	0.00	0.00	0.00	0.00
moving-bicyclist	0.01	0.00	0.17	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.01	0.00	0.18	0.00	0.00	0.00	0.17	0.00	0.02	0.00	0.00	0.02	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.01
moving-person	0.00	0.00	0.11	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.03	0.00	0.16	0.01	0.09	0.00	0.20	0.00	0.03	0.01	0.00	0.01	0.00	0.00	0.34	0.00	0.00	0.00	0.00	0.00
moving-motorcyclist	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.07	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.33	0.06	0.00	0.52	0.00	0.00	0.00	0.00
moving-on-rails	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
moving-bus	0.16	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.54	0.00	0.00	0.00	0.09	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.17	0.00	0.00
moving-truck	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
moving-other-vehicle	0.00	0.00	0.80	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00	0.00	0.00	0.06	0.00	0.00	0.00	0.00	0.00	0.09

Figure 5 – Confusion matrix

Given the unbalancing of the classes in the point clouds (see figure 6), the results of the confusion matrix (see figure 5) are understandable. For example, the persons are mostly confused with the vegetation, as there is more vegetation than persons in the data. Also, the 10 cm leaf size sub-sampling affected small objects.

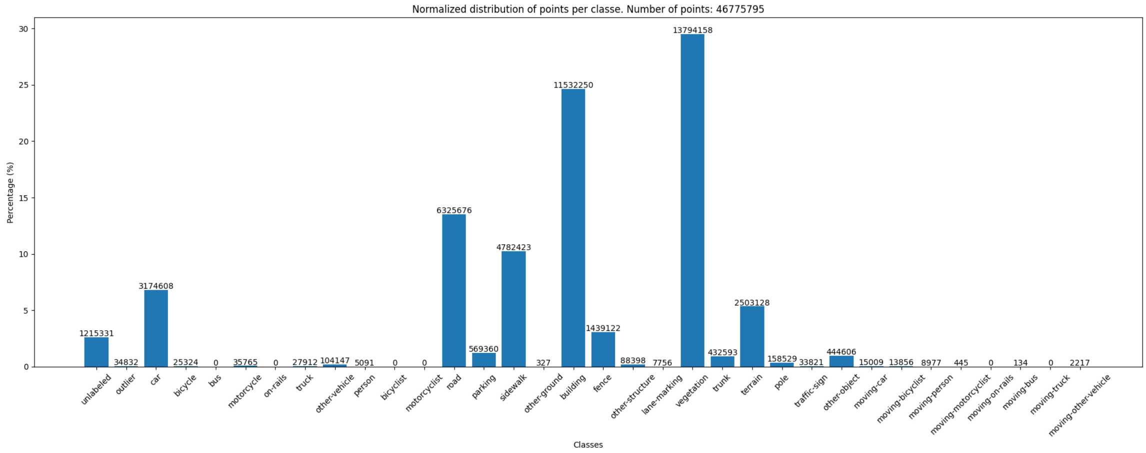
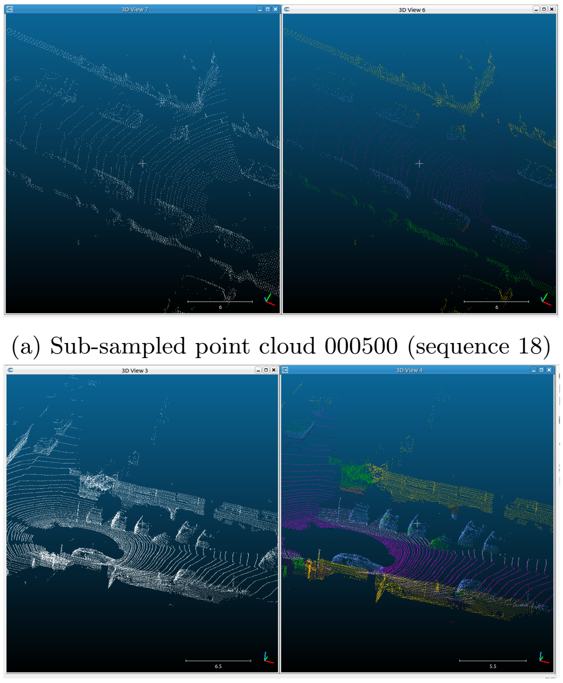


Figure 6 – Test set: Distribution of points per class

The trained model was used to predict the classes of a point cloud of sequence 18, which gave interesting results (see figure 7a).



(a) Sub-sampled point cloud 000500 (sequence 18)

(b) Sub-sampled point cloud 004540 (sequence 00)

Figure 7 – Semantic segmentation of point clouds

I also downloaded ParisLille dataset [4] and made a prediction on a sub-sampled slice of the Lille1 point cloud. Unfortunately, the model only predicted outliers (see figure 8).
This is because ParisLille dataset does not have the same distribution as kitti dataset. Furthermore, during training data processing I did not normalize the clouds into sphere unit as stated in the paper.

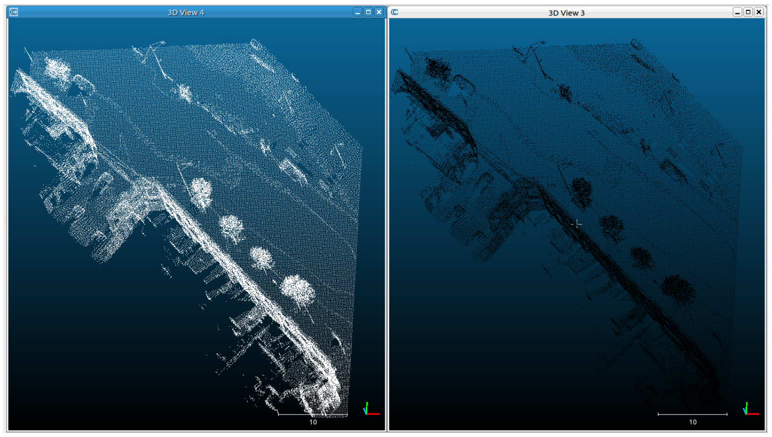


Figure 8 – ParisLille dataset - Prediction on a slice of Lille1 data before normalization

After normalizing the Lille1 3D point cloud according to the same distribution as the kitti dataset, the model can predict classes other than outliers for the ParisLille data (see figure 9). However, given the distorted appearance of the pointcloud after its normalization, the predictions on the ParisLille data are highly biased.

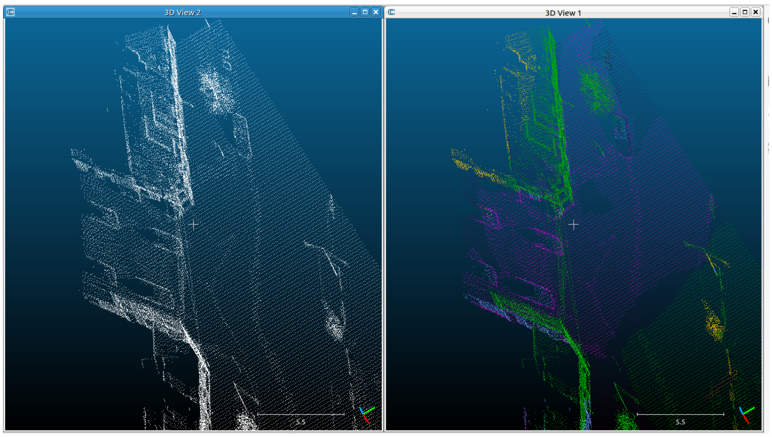


Figure 9 – ParisLille dataset - Prediction on a slice of Lille1 data after normalization

The model can be improved by performing during training, the data augmentation (rotation + jittering) as proposed in the original paper, and normalize the training points. It could also be interesting to merge the classes, for example a moving car remains a car.

4 References

[1] J. Behley et al. “SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences”. In: *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV).* 2019. URL: `http://www.semantic-kitti.org/` (visited on 11/25/2020).

[2] Andreas Geiger, Philip Lenz, and Raquel Urtasun. “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite”. In: *Conference on Computer Vision and Pattern Recognition (CVPR).* 2012. URL: `http://www.cvlibs.net/datasets/kitti/eval_odometry.php` (visited on 11/25/2020).

[3] Charles R Qi et al. “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”. In: *arXiv preprint arXiv:1612.00593* (2016). URL: `http://arxiv.org/abs/1612.00593` (visited on 11/25/2020).

[4] Xavier Roynard, Jean-Emmanuel Deschaud, and François Goulette. *Paris-Lille-3D: a large and high-quality ground truth urban point cloud dataset for automatic segmentation and classification.* URL: `https://arxiv.org/abs/1712.00032` (visited on 12/04/2020).