

Density-adaptive Sampling for Heterogeneous Point Cloud Object Segmentation in Autonomous Vehicle Applications

Hasan Asy'ari Arief

Norwegian University of Life Sciences
Ås, 1432, Norway
hasanari@nmbu.no

Manoj Bhat

Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA, USA
mbhat@andrew.cmu.edu

Håvard Tveite

Norwegian University of Life Sciences
Ås, 1432, Norway
havard.tveite@nmbu.no

Mansur Maturidi Arief

Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA, USA
marief@andrew.cmu.edu

Ulf Geir Indahl

Norwegian University of Life Sciences
Ås, 1432, Norway
ulf.indahl@nmbu.no

Ding Zhao

Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA, USA
dingzhao@cmu.edu

Abstract

Robust understanding of the driving scene is among the key steps for accurate object detection and reliable autonomous driving. Accomplishing these tasks with a high level of precision, however, is not trivial. One of the challenges come from dealing with the heterogeneous density distribution and massively imbalanced class representation in the point cloud data, making the crude implementation of deep learning architectures for point cloud data from other domains less effective. In this paper, we propose a density-adaptive sampling method that can deal with the point density problem while preserving point-object representation. The method works by balancing the point density of pre-gridded point cloud data using oversampling, and then empirically sample points from the balanced grid. Using the KITTI Vision 3D Benchmark dataset for point cloud segmentation and PointCNN as the classifier of choice, our proposal provides superior results compared to the original PointCNN implementation, improving the performance from 82.73% using voxel-based sampling to 92.25% using our proposed density-adaptive sampling in terms of per class accuracy.

1. Introduction

The ever-growing availability of large-scale point cloud data and easy access to affordable Light Detection and

Ranging (LiDAR) sensors have opened new streams of research in the computer vision field in recent years [10]. Instead of having to estimate distances between objects in a scene from images in order to reconstruct a 3D scene, researchers can now achieve the same objective by leveraging hundreds of thousands of point coordinates. PointNet [3] and PointCNN [13] are two popular deep learning models for learning the features of a scene from raw point cloud data, achieving an appealing performance of 83.7% for PointNet and 86.14% for PointCNN in terms of part-averaged IoU when applied on the ShapeNet Parts dataset [13]. Due to the novelty of its approach in accounting for the spatial information in the point cloud data, PointNet has commonly been used as the backbone of many recently developed deep learning models for raw point cloud data [17, 21].

Point cloud datasets usually contain an extremely large number of points, preventing deep learning models to include all the data points simultaneously. In that setting, it is common practice that only a few samples are considered in one learning iteration. The point sampling procedure in PointNet++, for example, is carried out by first partitioning the point cloud data into several voxels, and then performing random sampling from each of the voxels to construct training sets with the maximum number of points that is permissible by the architecture [18]. This approach presumes that the point density in each voxel is homogeneous. While the homogeneous point density distribution setting

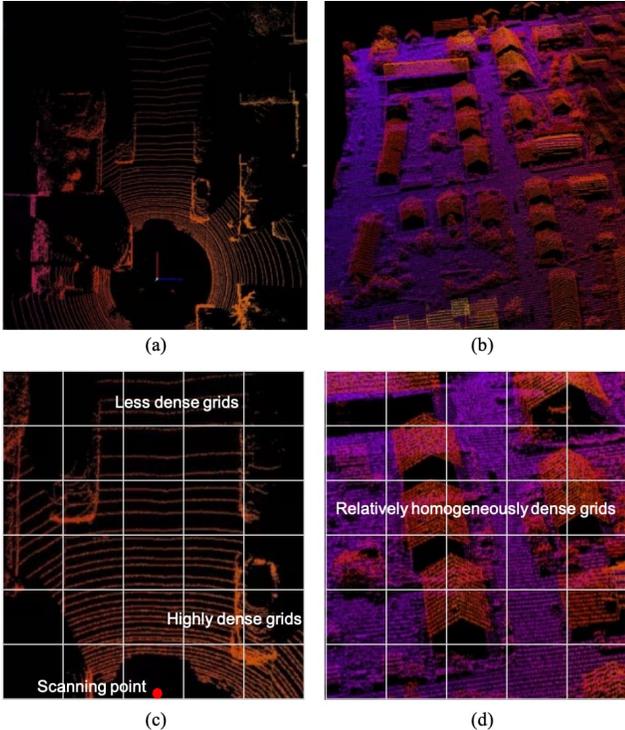


Figure 1. The nature of point cloud data from two different domains: (a) driving scene point cloud from Velodyne-type LiDAR (b) landscape map point cloud from airborne LiDAR along with point density distributions (c-d).

usually holds true for some applications, e.g. geomapping with point clouds from airborne LiDAR [1] or object scanning with point clouds from close-range scanning LiDAR [5], it is severely violated in the context of autonomous vehicle (AV) applications. In this context, the point cloud data are collected using a LiDAR with rotating beams (such as the Velodyne LiDAR) and the AV being in the center of the point clouds, thus the further the distance is, the less dense the points become (see for instance the Kitti dataset [9]). Because of this heterogeneity, the standard sampling technique with the voxelization method is not suitable. Fig. 1 illustrates the problem. In this figure, point cloud data from the Kitti Velodyne lidar (for a driving scene) and from the ISPRS Vaihingen airborne LiDAR (for a mapping application) are presented along with the top-view voxelization grids that are commonly used in training deep learning models.

The main contribution of our work is the use of a density-adaptive sampling method for improving the classification accuracy for heterogeneous point clouds with highly imbalanced class representation, especially when constructing the training set with an intention of balancing the point density distribution of the point cloud during the training process.

Other approaches have been studied in the context of learning from imbalanced datasets [2, 11]. We argue that obtaining point samples from grids of point cloud data with homogeneous point density distributions provides a good training set that is suitable with deep learning architectures utilizing raw point cloud data as input. In the experiment, we investigate the performance of various sampling strategies and observe that a class of sampling methods that are density-adaptive, i.e. taking into account the distribution of point density into the sampling scheme, yields a superior result compared to other sampling methods, including the crude voxelization-based sampling that is implemented in several existing deep learning architectures [13]. The experimental results using the Kitti 3D dataset show a significant improvement, from 82.73% using the original sampling to 92.25% using our proposed density-adaptive sampling, in terms of per class accuracy. We note that the performance of the proposed method is compared with accuracy metrics (true positive rate), rather than MIOU score, because our model discovered more objects compared to what the Kitti ground truth semantic labels provide, hence computing the MIOU score unfairly exacerbates our performance. A more detailed discussion will be provided in Section 5.

In what follows, we provide some background information describing the context of the challenges of work and review some related works. In Section 3, we present our method and the metrics used for evaluation. In Section 4, we describe the experiment settings and the Kitti dataset and show our results. In Section 5, we discuss our findings and finally we present the conclusion in Section 6.

2. Related Work

In this section, we provide an overview of the nature of point cloud data and an overview of selected work related to semantic segmentation using raw 3D point cloud data and sampling methods for learning models.

2.1. Point cloud and LiDAR

The growing interest analyzing point cloud data that capture spatial features of complex 3D scenes have become a trend in various fields, including mapping, object reconstruction, driving navigation, etc. A point cloud is a set of points in 3D space, each described by x , y , z coordinate values that are usually collected by LiDAR sensors. The sensor sends out pulses of beams at high frequency and calculates object distances based on the time for the pulses to return. By sweeping a region of interest with the LiDAR beams and combining the information from all points in a scene, usually in the order of hundreds of thousands or even a few million points, a data collection effort using LiDAR can provide accurate spatial information of complex scenes quite efficiently.

The point cloud data inherit unique characteristics, e.g. in terms of the distribution of point density in various sub-regions or grids of the scene or object of interest, depending on the nature of the scene of interest and the types of LiDAR utilized [16, 20]. Point cloud data from Airborne LiDAR, for instance, which is widely used for mapping geographical areas from a top-view, generally have a relatively even point density in each grid of the scene. Point clouds obtained from a close-range portable 3D object scanner LiDAR will have a point density distribution according to the surface captured by the scanning process. In contrast, point cloud data collected from Velodyne-type LiDARs, which is prevalent in AV applications with 360-degree surrounding environment as the scene of interest, will have highly heterogeneous point density in its grids. Fig. 1 shows the difference of point density distribution for different types of point cloud data.

2.2. Deep learning for point cloud segmentation

On its own, obtaining accurate semantic segmentation label from scenery data has less meaningful practical uses. However, semantic segmentation has been shown to be a good processing mechanism for object detection on point cloud data [6, 15]. That is, the scene segmentation task, which often is implemented as a supervised learning scheme, can be coupled with a classification or clustering method to build an accurate and efficient object detection pipeline or end-to-end framework. Hence, achieving a good performance for semantic segmentation on point cloud is a good intermediate objective. [21] provides a good review of the various frameworks used for semantic segmentation for point cloud data.

On a high level, these frameworks can be divided into several classes based on the input data that are exploited in the training of the deep learning models. Some frameworks [4, 14, 22] use an image projection or voxel representation of the point cloud data making it ready for the convolutional operations in the deep architectures. PointNet++ and PointCNN use the raw point cloud data directly which to some extent should have the advantage of being able to exploit the spatial information from the point cloud data. In these frameworks, the training set is constructed via sampling method. In the original PointNet++ implementation, the sampling is done by partitioning the scene of interest into several overlapping partitions by some distance metric, from which the local features are extracted. The local features are then clustered into larger units in a hierarchical fashion to capture the features of the whole scene [18]. PointCNN uses point cloud data and learns by utilizing the so-called X-Conv operations [13]. Similar to the convolution operation in ConvNet [12], X-Conv includes the calculation of inner products of transformed point cloud data and convolution kernels. The learning process relies on the

Multi-Layer Perceptron (MLP) algorithm and uses a Unet-like architecture [19] to do point-level segmentation. The implementation of both methods on the ShapeNet data set have yielded appealing results: 85.1% for PointNet++ and 86.14% for PointCNN. These methods, however, have not obtained the same performance when applied on point cloud data from Velodyne LiDAR, such as the Kitti dataset.

2.3. Sampling methods

Selection of the training data for training a deep learning model is always a critical task with respect to future good generalization. The large number of point clouds in a scene combined with the heterogeneous point density distribution and the implementation of complex deep learning architectures necessitate an efficient sampling scheme that is capable of selecting a good set of points that are informative for training purposes. In the machine learning literature, this problem is closely related to learning from imbalanced classes. Researchers have proposed the use of various sampling strategies, including random sampling, oversampling, undersampling, and stratified sampling, to achieve certain criteria that balance the training set [2, 7]. Interested readers are referred to [11] and [7] for more extended discussions about training set optimization for deep learning models.

3. Methodology

Our proposed method uses density-adaptive sampling that can be achieved by performing oversampling in grid cells where the number of points is below a certain threshold. In this section, we will elaborate how this density-adaptive sampling scheme is implemented as part of semantic segmentation framework to assist the learning models learn the most scene features from the training data.

3.1. Semantic segmentation framework

Semantic segmentation for point cloud data is essentially a point-wise classification task, where each point in the point cloud is classified into the class-object the point belongs to. In the AV context, point cloud segmentation is utilized to assign object class (such as car, pedestrian and cyclist) to each point and to use the segmented points for generating object bounding boxes. We used a step-wise semantic segmentation pipeline for Velodyne-based point cloud data by implementing a density-adaptive sampling technique in the preprocessing of the input data. We then employed PointCNN feature learning to generate probability maps. See Fig. 2 for an illustration of the framework.

3.2. Sampling method

To address the density problem, we utilize density-adaptive sampling method. The key idea is that feature learning using more balanced training sets will ease the

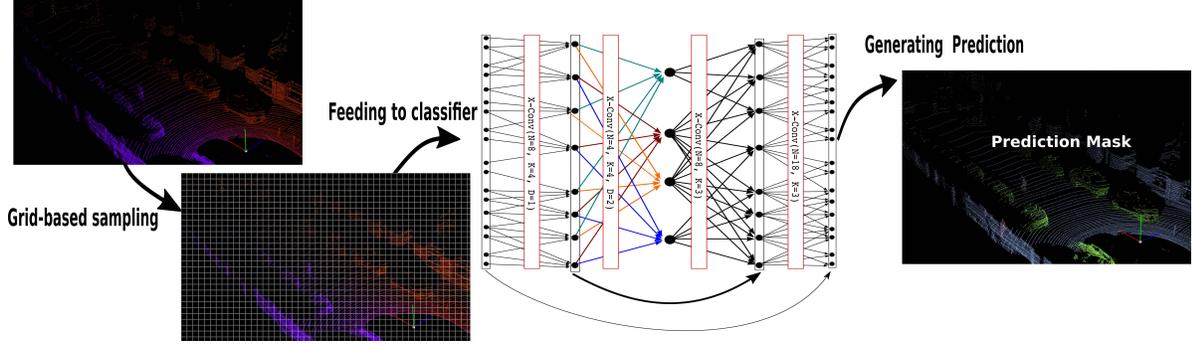


Figure 2. The proposed pipeline for semantic segmentation with PointCNN as the classifier of choice.

learning process of the deep learning kernels. In that sense, the density-adaptive sampling aims to amplify the likelihood that features from scenes with fewer points will be considered in the learning iterations. Density-adaptive sampling scheme can be achieved by using grid-based uniform sampling. This sampling method calculates the average point density (apd) on pre-gridded 3D point cloud data. The point density in each grid will then be normalized by over-sampling points within the grid cell to make the point density equal to the value of apd . Finally, an empirical or uniform sampling technique (without replacement) is applied to the normalized-density-grid to select the points used for training the deep learning model.

3.3. Classification algorithm

In this study, we use PointCNN as the classifier of choice. The novelty of PointCNN in learning from point cloud data is the use of nearby points as features for a point of interest. It uses point cloud as input represented as (x, y, z) coordinates along with a scalar denoting the intensity value for each point. With these inputs, PointCNN enriches the features by exploiting the spatial auto-correlation of nearby points. Technically, PointCNN employs hierarchical convolution; this feature is similar to the well-known pooling layer in ConvNet. The hierarchical convolution of PointCNN aggregates information from neighborhood feature maps to fewer points by applying the X-Conv operation recursively, clusters nearby points as the feature representation of the point of interest using a K -nearest neighbor algorithm, and projects the clustered points into the local coordinate system with the point of interest being the center of the cluster. After a series of transformations based on the PointCNN pipeline using MLP coupled with X-Conv operators and higher-dimensional projections, the segmentation class for each point is acquired.

3.4. Evaluation metric

For a class-imbalanced data set, such as AV point cloud data where the environment class by far dominates the in-

teresting classes (such as car, pedestrian, cyclist, etc.), only a few alternative metrics will be appropriate for evaluating the segmentation task. This is the case because metrics such as the overall accuracy performance are meaningless. By just ignoring building a classifier and instead using the simple rule of assigning all points into environment class would give good results in terms of the overall accuracy. To avoid this issue, we use Mean of Per-class Accuracy (MPA) metric as a notion of accuracy. The simultaneous comparison of true positive (TP) rate as correct prediction and false negative (FN) combined with false positive (FP) rate as wrong prediction is a common practice. Hence, we will also consider the Mean Intersection over Union ($MIoU$). The calculation of each metric is as follow, with k being the total number of classes and p_{ij} being the number of points of class i but classified as class j .

$$MPA = \frac{1}{k} \sum_{i=0}^{k-1} \frac{p_{ii}}{\sum_{j=0}^{k-1} p_{ij}}, \text{ and} \quad (1)$$

$$MIoU = \frac{1}{k} \sum_{i=0}^{k-1} \frac{p_{ii}}{\sum_{j=0}^{k-1} p_{ij} + p_{ji} - p_{ii}}. \quad (2)$$

4. Data and Experiment

We use the KITTI Vision Benchmark 3D data set [8] for our experiment, which records point clouds of driving scenes from Karlsruhe, Germany. The Kitti point cloud data set is collected using a Velodyne HDL-64E rotating 3D laser scanner, collecting data with 64 beams at 10 Hz. For our experiment purposes, we downloaded the 3D point cloud data set and labels from the Kitti website, totaling 29 GB in size. The data set contains 7481 scenes from an ego car viewpoint located in the center of the scenes. On average, each scene has 1.3 million points.

The labels provided are in the form of bounding box coordinates and class label (e.g. car, pedestrian, cyclist, etc.) for each of the boxes. We treat any points outside the bounding boxes for the pedestrian, car, and cyclist classes

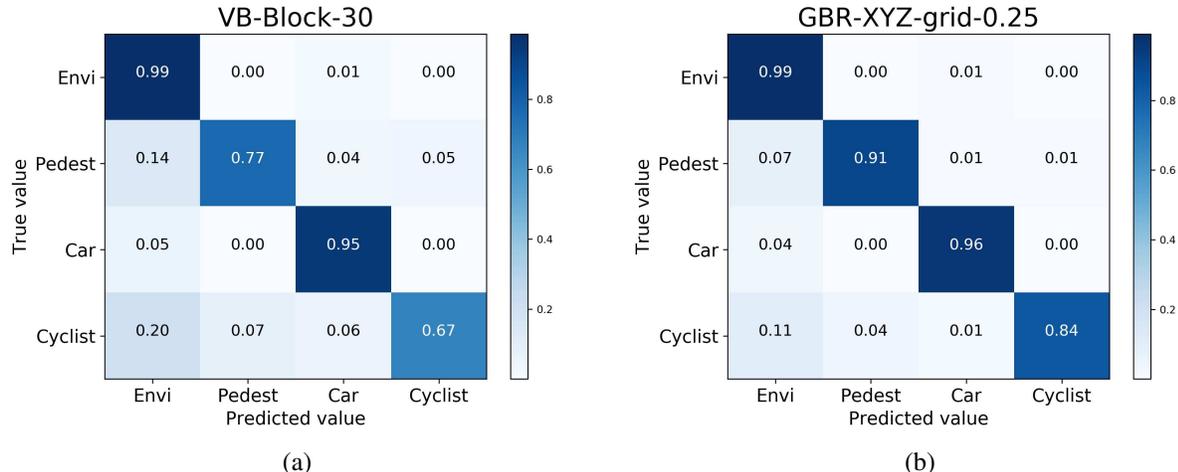


Figure 3. The accuracy performance (per-class accuracy) for the trained model using (a) the original voxel-based sampling and (b) the best-performing density-adaptive sampling.

Class	Number of Points (Percentage)	
	Training Data	Validation Data
Environment	604,128,641 (98.46%)	273,993,242 (98.44%)
Pedestrian	561,147 (0.091%)	271,529 (0.10%)
Car	8,705,256 (1.418%)	3,990,825 (1.43%)
Cyclist	181,974 (0.029%)	87,222 (0.03%)

Table 1. The distribution of class in the data set.

as environment, so most of the points belong to the environment class (around 98% of all the points). The distribution of classes is shown in Table 1. Moreover, the point density is decreasing with distance. To set the stage, we preprocessed the data by assigning the class label of the box to all points it contains, so we obtained point-wise labels. We then used 5,145 scenes for training and 2,336 scenes for validation purposes.

4.1. Experimental setup

Each PointCNN model was trained using one 11 GB GeForce GTX 1080 Ti graphics card with eight-point blocks per batch, following the capacity limitation of the GPU. The Tensorflow version of PointCNN was used as the training code and environment. Unless otherwise noted, initial learning rate for all models is 0.005 with 20% learning rate decay for every 5,000 iterations and was trained for 125,000 iterations. In order to force the model to recognize objects during training for such an imbalance data set, the weighted penalty (for loss calculation) for the environment class was set to 0.1, and to 1 for other classes. For calculat-

Method	<i>MPA</i>	<i>MIoU</i>
VB Block-10	0.6329	0.3369
VB Block-20	0.6999	0.5263
VB Block-30	0.8273	0.5717
Without oversampling		
GBR-Original	0.8895	0.6418
GBU-Original-XY-0.25	0.8641	0.6179
GBU-Original-XY-1	0.8731	0.6511
XY-GRID with Oversampling		
GBU-XY-grid-0.25	0.8876	0.5840
GBU-XY-grid-1	0.8983	0.6304
GBR-XY-grid-0.25	0.9152	0.6342
GBR-XY-grid-1	0.8881	0.6504
XYZ-GRID with Oversampling		
GBU-XYZ-grid-0.25	0.9026	0.6471
GBU-XYZ-grid-1	0.9040	0.6346
GBR-XYZ-grid-0.25	0.9225	0.6816
GBR-XYZ-grid-1	0.9217	0.6509

Table 2. The performance of each of the sampling scenario.

ing the validation accuracy, the weighted penalty for all the classes was set to 1. The highest *MPA* and *MIoU* score for the validation data are then used to evaluate and compare the prediction accuracy for all models in the experiment.

4.2. Sampling scenario

We test our hypothesis by training the PointCNN with different sampling methods. The benchmark is the voxel-based sampling (the original PointCNN sampling package). We also include grid-based uniform sampling and grid-based random sampling for comparison. Each of the methods sample (without replacement) 10,000 points per block

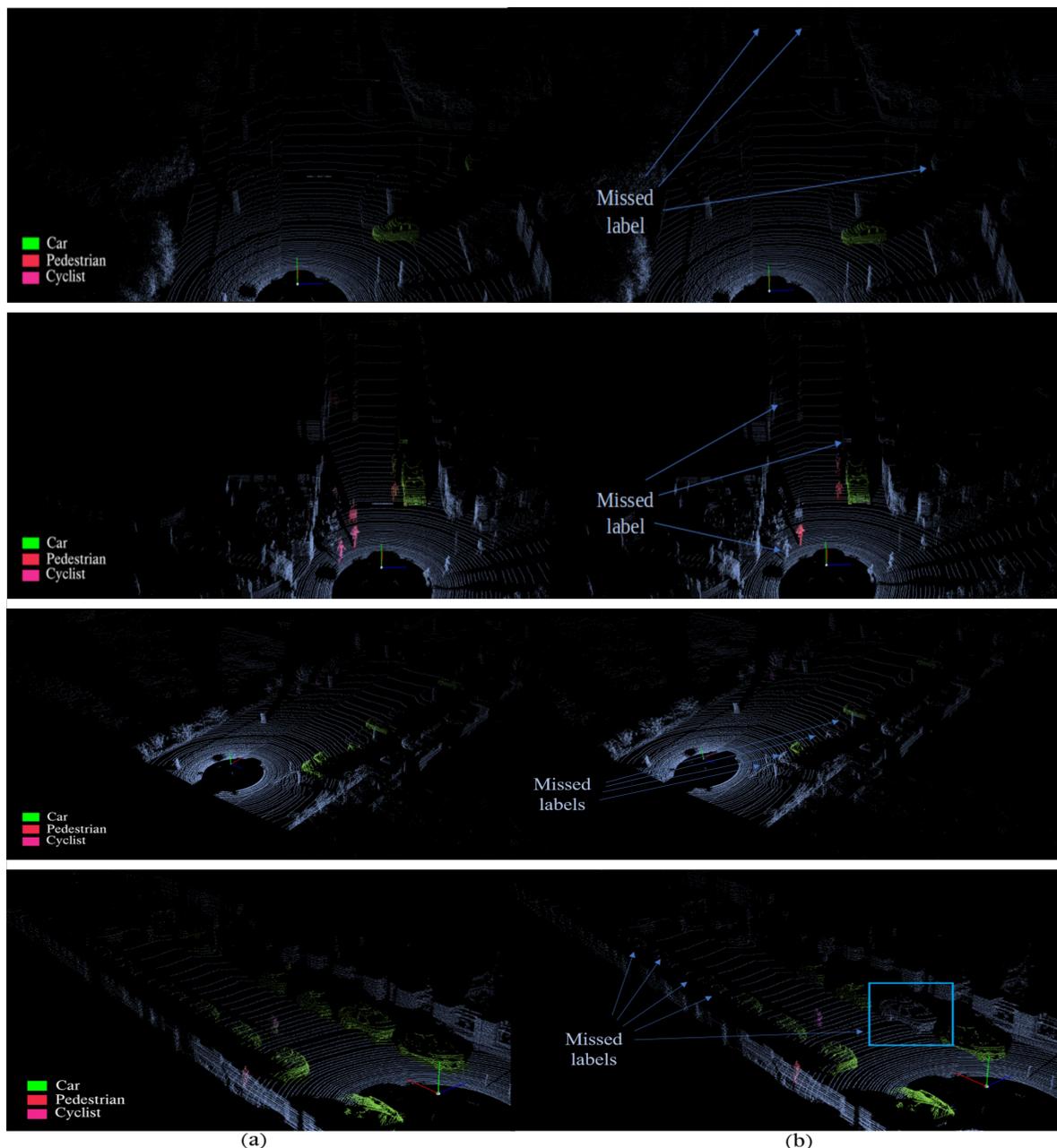


Figure 4. Point cloud visualization with (a) Prediction results, and (b) Ground truth label with missing object bounding box.

to be used as the PointCNN input. PointCNN, for every training iteration, will sample 2,048 random points to be used. It should be noted that randomly selecting 2048 points from 10,000 points per iteration will generate different point sets, hence increasing the variation of data learned by the PointCNN model.

In terms of sampling parameters, block sizes of 10, 20, and 30 coordinate values were used for the voxel-based sampling methods with the assumption that the average

point densities per block are 100, 25, and 10, respectively. The grid-based sampling methods used grid size 0.25 and 1 coordinate values. For the sake of simplicity, we refer to the voxel-based method as VB, grid-based uniform sampling as GBU, and grid-based random sampling as GBR throughout this paper. We also include versions of GBU and GBR without the rebalancing point density and call these versions GBU-Original and GBR-Original, respectively. In addition, we also compare the results for 2D and 3D grids

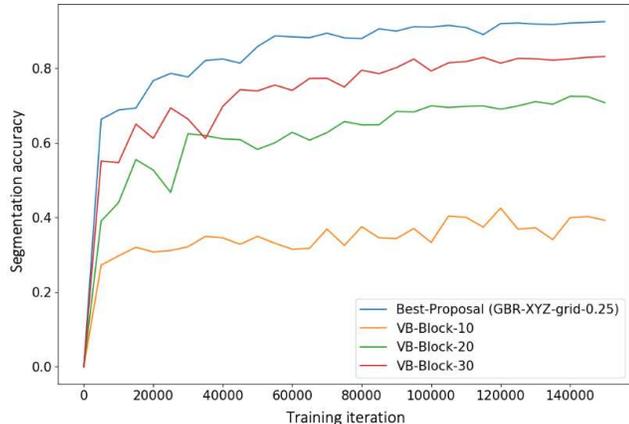


Figure 5. The *MPA* trends of the model during training using various sampling schemes.

(*XY* and *XYZ* axis) for both *GBU* and *GBR* sampling. Table 2 shows the performance of the resulting models in terms of *MPA* and *MIoU*. Fig. 5 shows the per-class accuracy using the original voxel-based method and the best-performing density-adaptive sampling method.

5. Discussion

In this section, we discuss our findings related to the accuracy, learning stability, and the *MIoU* performances.

5.1. Per-class accuracy performance

As shown in Table 2, the proposed method, using a density-adaptive sampling scheme, yields superior mean per-class accuracy performance compared to the original voxel-based sampling method. The best performance for the proposal achieves 92.25% (*GBR-XYZ-grid-0.25*) compared to 82.73% for the original sampling method (*VB-Block-30*). Table 2 shows the voxelization in the original sampling scheme does not improve the accuracy performance. It suggests that removing the partitions of the training sets while keeping the heterogeneous density, thus shifting from stratified sampling into randomized sampling based on the empirical distribution of the original data, only alleviates a part of the problem.

On the other hand, density-adaptive sampling methods, which superficially augment the representativeness of the features from all grids in the scene by means of oversampling, seems effective in increasing the generalizability of the trained model when classifying the unseen data, and works best when coupled with randomized sampling and a small grid size. The best-performing grid size for *GBR-XYZ* density-adaptive sampling class is 0.25, which is reasonable, because the smallest grid size forces the training points to be most evenly distributed covering whole scene.

Finally, Fig. 3 presents the confusion matrix showing *TP*, *TN*, *FP*, and *FN* for PointCNN trained using the original sampling method (top) and our density-adaptive method (bottom). It is clear that the proposed sampling method attains a model with better accuracy performance for all classes. The most significant increase in performance is observed for classes with fewer points (pedestrian and cyclist), thus indicating the robustness of our sampling scheme in dealing with class-imbalanced data sets.

5.2. Performance stability and learning rate

In Fig. 5, the accuracy of the model during training is computed against the validation data set in each training iteration. Notice that the model trained using the original voxel-based sampling method with grid size 10 (*VB-Block-10*) has the most volatile performance and the other original *VB* methods have less volatile performance in earlier stages of its learning procedure, while the density-adaptive methods have more stable performance during training. This observation indicates that the training set from density-adaptive methods makes it easier for the deep learning models to learn hidden features from the point cloud data, hence achieving more stable learning rate and faster convergence. It is therefore promising to devise an objective function and parameterize the sampling scheme so as to maximize the learning rate and overall performance of the deep learning model in this setting. We plan to derive such a schema in our future work.

5.3. The *MIoU* and inaccurate bounding boxes

Fig. 4 shows the comparison of our segmentation result (left) with the provided labels from the *Kitti* data set (right), where the trained model discovers more car objects than the ground truth label provides (the car highlighted in the right figure is not labeled). We also notice some inaccurate bounding box locations. Computing the *MIoU* metric penalizes the trained model for discovering unlabeled objects or locating the bounding boxes more accurately, due to the smaller intersection (appearing in the numerator of Equation 2) and larger union (in the denominator of Equation 2) between boxes from the provided labels and model prediction. In this situation, we argue that the *MIoU* cannot be used as an appropriate metric for segmentation performance. Therefore, we cannot use the *MIoU* from Table 2 to compare the performance of the sampling methods, though it is interesting to see that the *MIoU* score from the original voxel-based sampling scheme.

6. Conclusion

In this study, we have proposed a density-adaptive sampling method to be combined with a deep learning architecture for semantic segmentation using raw point cloud

data, such as the PointCNN. We have shown that the heterogeneous point density and the class imbalance of the point cloud from Velodyne-type LiDARs in AV applications render the voxel-based sampling method in the original PointCNN implementation ineffective for constructing a good training data set. By implementing density-adaptive sampling, i.e. by oversampling the less dense grids of the scene hence augmenting the representativeness of the feature in the less dense scenes using various parameterization scenarios, we show that the deep learning models yields superior performance in terms of classification accuracy and learning stability. Furthermore, we have shown empirically that the trained model is more robust against the class imbalance that is prevalent in real-world driving scenes.

Acknowledgment

This research was funded in part by gifts from Bosch Corporate Research. The authors would like to thank Ji Eun Kim and Wan-Yi Lin for their valuable discussion and suggestions.

References

- [1] Hasan Asyari Arief, Geir-Harald Strand, Håvard Tveite, and Ulf Indahl. Land cover segmentation of airborne lidar data using stochastic atrous network. *Remote Sensing*, 10(6):973, 2018. 2
- [2] Gustavo EAPA Batista, Ronaldo C Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1):20–29, 2004. 2, 3
- [3] R. Qi Charles, Hao Su, Mo Kaichun, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1
- [4] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017. 3
- [5] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Niebner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2
- [6] Bertrand Douillard, James Underwood, Noah Kuntz, Vsevolod Vlaskine, Alastair Quadros, Peter Morton, and Alon Frenkel. On the segmentation of 3d lidar point clouds. In *2011 IEEE International Conference on Robotics and Automation*, pages 2798–2805. IEEE, 2011. 3
- [7] Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational intelligence*, 20(1):18–36, 2004. 3
- [8] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 4
- [9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012. 2
- [10] Mark Harris. Ces 2018: Waiting for the \$100 lidar, Jan 2018. 1
- [11] Julio Isidro, Jean-Luc Jannink, Deniz Akdemir, Jesse Poland, Nicolas Heslot, and Mark E Sorrells. Training set optimization under population structure in genomic selection. *Theoretical and applied genetics*, 128(1):145–158, 2015. 2, 3
- [12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):8490, 2017. 3
- [13] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*, pages 820–830, 2018. 1, 2, 3
- [14] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. Deep continuous fusion for multi-sensor 3d object detection. *Computer Vision ECCV 2018 Lecture Notes in Computer Science*, page 663678, 2018. 3
- [15] Tomasz Malisiewicz and Alexei A Efros. Improving spatial support for objects via multiple segmentations. 2007. 3
- [16] Anh Nguyen and Bac Le. 3d point cloud segmentation: A survey. In *2013 6th IEEE conference on robotics, automation and mechatronics (RAM)*, pages 225–230. IEEE, 2013. 3
- [17] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018. 1
- [18] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017. 1, 3
- [19] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 3
- [20] Brent Schwarz. Lidar: Mapping the world in 3d. *Nature Photonics*, 4(7):429, 2010. 3
- [21] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. Pointcnn: 3d object proposal generation and detection from point cloud. *arXiv preprint arXiv:1812.04244*, 2018. 1, 3
- [22] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018. 3