

ADVERSARIAL NETWORK FOR LIDAR POINT CLOUD SEGMENTATION

Jheng-Lun Liou (劉政倫)¹, Augustine Tsai (蔡岳廷)^{2*}, Chiou-Shann Fuh (傅楸善)³,
and Fay Huang (黃于飛)⁴

^{1,3}Department of Computer Science and Information Engineering,
National Taiwan University, Taiwan

²Institute for Information Industry, Taiwan

^{2,4}Department of Computer Science and Information Engineering,
National Ilan University, Taiwan

*E-mail: atsai@niu.edu.tw

ABSTRACT

LiDAR has been serving as a de facto perception subsystem for autonomous vehicles (AV). The point cloud map captured by LiDAR offers a rich semantic information for AV to navigate safely. In this paper, we extend the 2D adversarial segmentation network for 3D point cloud processing. The adversarial discriminator is used to improve the spatial continuity and label consistency in segmentation, it is only active in network training and is dropped during the inference. Both qualitative and quantitative measurements are conducted on KITTI dataset.

Keywords: LiDAR, Autonomous Vehicle, Point Cloud Segmentation, Adversarial Network.

1. INTRODUCTION

Environmental semantics is important for autonomous vehicle (AV) navigation. The ability to comprehend the drivable areas and recognition of dynamic objects are critical to safe manoeuvre in urban traffic. LiDAR is now the de facto perception subsystem for most of the AVs. Its active sensing capabilities can precisely return surrounding object distances for AV motion control and planning, and is also immune to the lighting changes.

In this paper, we concentrate on surrounding point cloud segmentation using data collected by the 360 degree spinning LiDAR. Traditional methods [1] for scene understanding comprises multiple stages, such as ground removal, remaining point cloud clustering. Apply hand-crafted features on the cluster for instance recognition. Many of the ground removal algorithms depend on normal vectors and iterative RANSAC (random sample consensus)

In this paper, we proposed a one stage deep learning approach including feature extraction and final segmentation.

2. RELATED WORK

There are two camps of deep learning based point cloud segmentation. One is to process point cloud directly in 3D domain, such as PointNet [2], the other one is to project point cloud onto a surrounding spherical plane, and the result is a 2D range imagery. Therefore, many 2D segmentation techniques can be readily applied.

Some of early works based on 2D projection segmentation were proposed by Wu et. al. [3,4,5], they used a light weighted SqueezeNet which has reduced parameters and low computation complexity, as well as fast realtime inference. SqueezeNet is used for initial segmentation, and then a reformatted CRF (conditional random field) [5] as RNN (recurrent neural network) is concatenated for post processing.

In 2D segmentation, CNN based architecture predicts each label class independently, so post processing such as CRF is required to reinforce the spatial continuity for the output label maps. However, CRF is limited to the optimization of unary and pairwise potentials. Higher

order potentials has been shown to be effective for super-pixel label consistency. Recently, adversarial network [6.7] is shown to be efficient as alternative to CRF. The main contribution of this paper is to extend the adversarial 2D segmentation for 3D point cloud and incorporate a more computational efficient loss function.

3. PROPOSED APPROACH

3.1. Spherical Projection

In order to readily apply 2D segmentation technique, 3D point cloud first converted to a 2D imagery by spherical projection. This process converts continuous information of angles into discrete information of 2D planar pixels. For implementation, collection frequency of our LiDAR is set to a line sweep to get 2,048 points, so a 64-line LiDAR will emit a total of $2,048 \times 64$ points. Next, we can intuitively regard $2,048 \times 64$ as 2D x, y coordinates. Now, we only need to quantify the 2D coordinates corresponding to each point-cloud data, and then put the depth and intensity values collected by LiDAR into the correct pixels.

Point cloud data collected from LiDAR is originally stored in 3D Cartesian coordinates. Therefore, the depth or distance of each 3D point can be obtained by the standard distance formula:

$$distance = \sqrt{x^2 + y^2 + z^2}$$

In some cases, the reflection values of the emitting rays cannot be received by the sensor, the distance values for such corresponding image pixels are therefore undefined.

Altogether 5 values are appended to each 2D x, y coordinate position. They are the associated 3D Cartesian coordinates, the intensity value collected by the sensor, and the distance (if available) calculated by the above formula. Eventually a $2,048 \times 64 \times 5$ data matrix can be obtained and sent to a CNN for 2D segmentation.

3.2. Overall Segmentation Model Architecture

After considering many state-of-the-art techniques and taking large number of trials and errors, a suitable loss function was proposed in the end. The general goal is to achieve high compression rate of the parameters and as well as high segmentation accuracy. The proposed conditional generative adversarial learning architecture can improve correlation between pixels that are often ignored in semantic segmentation prediction results.

3.2.1 Adversarial Training and Model Architecture
CGAN (Conditional Generative Adversarial Network) [9] adds conditional constraint to GAN [8]. A CGAN-based 2D segmentation architecture has been proposed and illustrated in Figure 3.2.1(a), ~~which has been referred to as MamboNet~~. Three important images in this Network represent Input Data, Predicted Label, and Ground Truth Label, respectively. Input Data are input generator data and our CGAN conditions. Predicted Label is the output of the generator: fake data in GAN. Moreover, Ground Truth Label is the ground truth data in GAN.

In Figure 3.2.1(a), Predicted Label and Ground Truth Label were sent to the discriminator after individually concatenating condition constraints of Input Data to them. Imposing constraints is the essential idea of CGAN. However, applying CGAN to the segmentation network and defining the concepts of Ground Truth Label, Predicted Label, and Input data are innovative ideas of our propose network.

The training method has three major benefits: First, it allows generator to learn correlation between neighboring pixels, which appears to be the most common problem in pixel-level segmentation networks. Although CNN structure in the encoder takes local features into consideration, the procedures in decoder tend to lose information over the up-sampling processes. For example, we may compare the Predicted Label of Figure 3.2.1(b) without CGAN and the Predicted Label in the proposed network with CGAN in Figure 3.2.1(a).



Figure 3.1 Images of point cloud data collected by Ouster OS-128 after coordinate transformation. Euclidean distance of each 3D point is represented by various intensity value. Note that the black image regions indicate the areas where the distances are undefined, such as the sky.

Only the front part of the truck has been correctly identified if processed by a general segmentation network as shown in Figure 3.2.1(b). The rear part of the

truck was misidentified as car. The problem can be overcome using the proposed network along with the CGAN concept. More supporting results will be shown

in the Experimental Evaluation section. The second benefit is that the network only increases calculation overhead of discriminator during training. GAN aims to allow discriminator to influence generator, so that generator can be trained through CGAN principle to consider neighboring pixel correlation. Although discriminator parameters are still needed during training, we not only need segmentation results of generator during inference, but also do not have pre-labeled Ground Truth Label during inference. Thus, we only need generator during inference. Furthermore, the proposed architecture can drop discriminator during actual inference. Compared with the original segmentation network, new network does not increase the number of parameters. Thus, our segmentation model achieves

comparable performance with the state-of-the-art approaches with low number of parameters for self-driving car applications. Finally, the third benefit is that its discriminator is pluggable everywhere. Proposed network enhances CGAN with more advanced segmentation network architecture. After segmentation network architecture as generator, we only need to plug in discriminator to increase accuracy. For example, with segmentation network achieving the tightest and most excellent segmentation on limited hardware resources, we can increase accuracy by plugging in our discriminator for training. Only GPU (Graphics Processing Unit) during training requires more memory, and inference does not consume any additional memory at all.

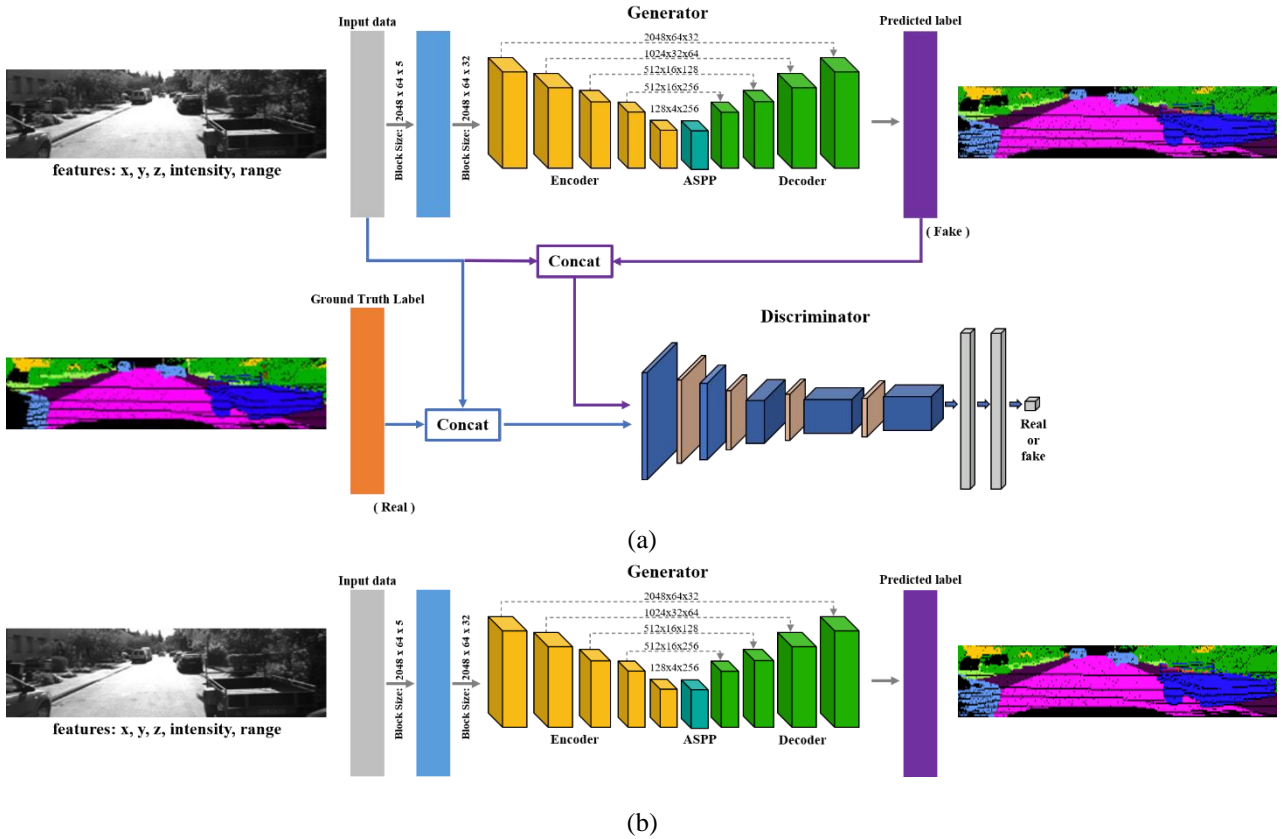


Figure 3.2.1 Segmentation network of (a) the proposed architecture and (b) the conventional network. In the resultant images, the category of car is represented by light blue and truck is represented by dark blue.

3.2.2 Generator

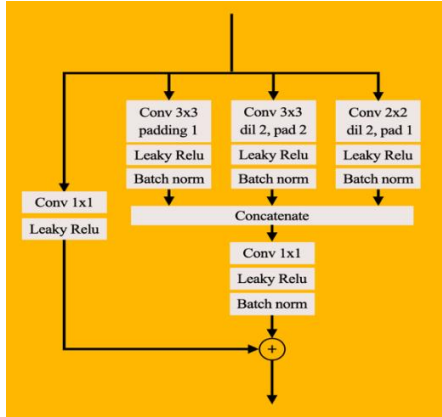
The generator is based on CNN model, which consists of three parts: encoder, decoder, and ASPP, as shown in Figure 3.2.1 (a). Many advanced technologies have been taken into consideration. Encoder part uses Resnet [11], Dilated kernel [12], MobileNets [13], Inception Block [14, 15], ResNext [16], SE block [17] and so on. Resnet solves the problem of gradient explosion by transmitting the residuals additively and makes deep learning possible. Dilated kernel increases receptive field. Pointwise

convolution helps our model to reduce parameters without affecting efficiency. Then use property of Inception Block for integration, characteristic of split-transform-merge is claimed in ResNext [16]. Figure 3.2.2 (b) is the basic structure of our Convolution Block each time. SE (Squeeze and Excitation) Block aims to perform attention-related operations on each feature map and gives higher attention to feature maps with higher learning weights.

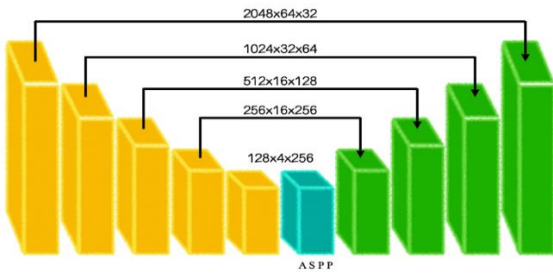
Decoder mainly uses Pixelshuffle [18] for Super Resolution as main up-sample method. To solve information loss problem that encoder down-sampling cannot be restored in decoder, encoder information is also sent to the decoder. Then several layers of CNN in decoder are used to effectively learn the transmitted information. Average Pooling is applied before sending, because Average Pooling can store more information. Max Pooling can get more significant features for classification. Besides, in Figure 3.2.2(a) [19], dilated kernel can increase receptive field but may create the gridding effect. Therefore, ASPP [20] aims to solves the effect with hole convolution of different rates, and finally connects hole convolutions of different rates to form a pyramid structure. Finally, the overall generator network architecture is shown in Figure 3.2.2(c).



(a) Problems of dilated kernel.



(b) Segmentation encoder of Generator.



(c) Generator architecture.

Figure 3.2.2 Generator model architecture in detail.

3.2.3 Discriminator

Discriminator is a 2D classification network to distinguish between real data and fake data. Here we regard Ground Truth Label as real data, and Predicted Label as fake data generated by generator. The biggest difference is that Predicted Label does not consider the correlation between pixels. In Figure 3.2.1 (b), if the

discriminator is not added, the segmentation network is likely to happen that the front half of the entire truck is recognized as a car, but the rear half is recognized as a truck, without considering pixel correlation. This type of example is also the biggest difficulty of pixel-level segmentation problem. To solve this difficulty, we consider local feature characteristics from CNN convolution. We hope that convolutional neural network will consider local pixel characteristics correlation to build the discriminator, and then can distinguish difference between Predict Label and Ground Truth Label, and make the generator use this as a starting point to improve. Thus, discriminator uses the simplest VGG (Visual Geometry Group) architecture in Figure 3.2.3. We do not specifically tune how many layers or some convolutional parameters are needed in this architecture, because we mainly want to use CNN to consider local features characteristics, so that entire GAN can generate segmentation results considering pixel correlation.

First, our discriminator can indeed obtain high accuracy in distinguishing true and false data. Second, during adversarial learning between discriminator and generator, generator does generate predictive labels that focus more on the correlation between pixels in Figure 3.2.1(a). In each epoch, discriminator can better distinguish the part that Predicted Label does not consider pixel correlation. Whenever discriminator discriminates better, under adversarial learning, generator will try its best to generate a Predicted Label with more pixel correlation in next epoch.

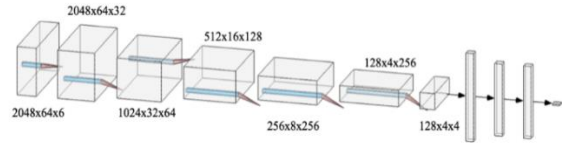


Figure 3.2.3 VGG-based Discriminator.

3.3. Loss Function

The loss function combines several different functions. The overall view can be divided into two categories: loss function of adversarial learning and loss function of segmentation network. First, CGAN adversarial loss function includes two architectures: generator and discriminator. Discriminator uses cross entropy, and generator also has CGAN adversarial function component. Generator is actually segmentation network architecture. Generator loss function includes three functions: related function of generating adversarial learning network, weighted cross entropy, and RMI (Region Mutual Information) Loss [21] that can focus more on local relevance.

3.3.1 Adversarial Functions

GAN performs mathematical problem:

$$\min_G (\max_D (\text{Objective Function}))$$

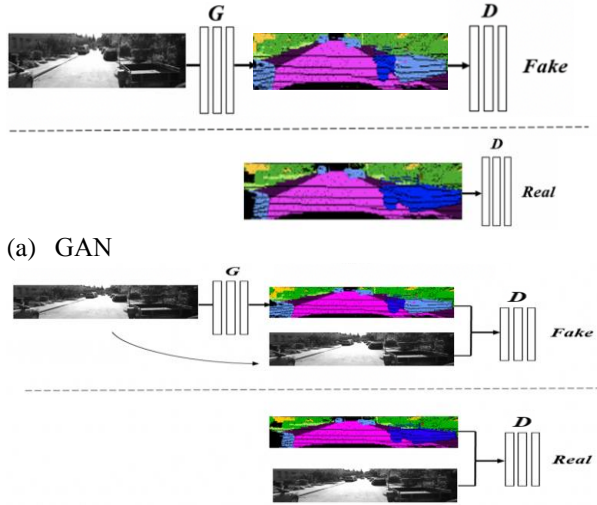
Function $\max_p(\text{Objective Function})$ actually trains discriminator, and obtain JS divergence of two data distributions. Function \min_G actually narrows JS divergence. The most primitive GAN target function:

$$E_{x \sim P_{GT(x)}}[\log D(x)] + E_{z \sim P_{G(z)}}[\log(1 - D(G(z)))]$$

Our CGAN objective function:

$$E_{x \sim P_{GT(x)}}[\log D(x|y)] + E_{z \sim P_{G(z)}}[\log(1 - D(G(z|y)))]$$

Objective function difference between GAN and CGAN is only one y : limitation to true and false information. CGAN optimization aims to train discriminator to obtain JS divergence between real data and fake data under condition y , and train generator to minimize JS divergence between the two under condition y : shape of Input Data. We need to limit Ground Truth Label, real data, and Predicted Label output by generator: fake data, to shape of Input Data for adversarial learning in Figures 3.2.1(a) and 3.3.1, as long as we concatenate condition to add. General GAN training is in Figure 3.3.1(a), and CGAN training is in Figure 3.3.1(b). The only difference is that condition y needs to be added before training.



(a) GAN

(b) CGAN
Figure 3.3.1 Training method compared with CGAN and GAN.

3.3.2 Segmentation Functions

First functions related to segmentation is Weight Cross Entropy (WCE) proposed by Salsanet [22]. Function $f(x, y)$ represents class frequency of pixels on x, y coordinates in overall point cloud. Weight $w(x, y)$:

$$w(x, y) = \frac{1}{\sqrt{f(x, y)}}$$

In Figure 3.3.2(a) [24], number of point clouds in different categories of SemanticKITTI is also very different. For example, whether it is SemanticKITTI or

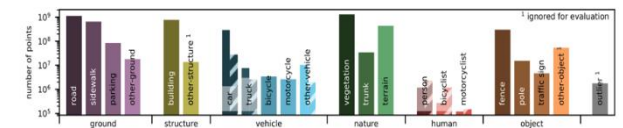
real-world street scenes, there will be more point clouds in the ground category than in human category. It is unfair to train them with the same CE weight. Therefore, WCE weight here considers each class frequently of the point cloud to solve the problem of unbalanced point cloud categories. Overall WCE loss function:

$$L_{WCE} = - \sum_{x,y} w(x, y) p(x, y) \log(q(x, y))$$

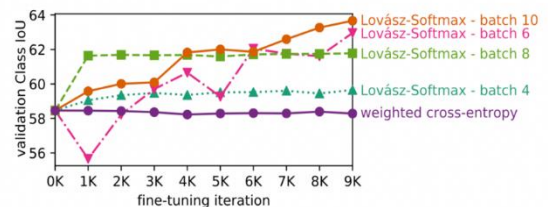
Generally, RMI [21] loss is often changed to [25] lovasz softmax, loss function used by most state-of-the-art semantic segmentation. However, in Figure 3.3.2(b) [25], lovasz softmax will affect performance at relatively low batch size. If GPU memory cannot load such high batch size, it will inevitably require other effective semantic segmentation loss functions that do not require much memory. Currently, we find state-of-the-art paper of Cityscapes, a semantic segmentation database that was also related to street scenes at the time, and find RMI Loss is used. Although Cityscapes is a database of RGB images, we have projected to 2D, and Cityscapes images are also street scenes with high reference value. Furthermore, RMI Loss mainly solves segmentation that often ignores the dependency between neighboring pixels. After using RMI Loss, we can get good results even in the same batch size compared with Lovasz Softmax. Finally, MamboNet, with only 11 GB of GPU Memory, combines various adversarial learning techniques and segmentation network model technology, and finally can surpass Salsanet in SemanticKITTI contest, which is equipped with huge GPU memory and uses Lovasz softmax as the loss function.

Now, we can know all the elements of generator loss function. Then, we can get the overall loss function of the generator as segmentation network:

$$L_{Generator} = \min_G(\max_D L_{CGAN}) + L_{RMI} + L_{WCE}$$



(a) Total number of point clouds in each category of SemanticKITTI. It shows data imbalance, and we can also know why we need to use WCE.



(b) Relationship between Lovasz softmax and batch size.

Figure 3.3.1 Schematic diagram of knowledge related to segmentation function.

4. EXPERIMENTAL EVALUATION

4.1. Semantic KITTI Dataset

In the vision of a self-driving car, LiDAR provides accurate geometric information about the environment, while the KITTI Dataset provides a large-scale data set of mobile LiDAR. Based on the need for automatic driving to understand the fine-grain of nearby targets and surfaces, SemanticKITTI is a label data set that has all 22 sequences of KITTI Vision Odometry Benchmark marked point by point. In Figure 4.1, categories labeled by SemanticKITTI are divided into 28 categories, of which 6 categories belong to mobile or non-mobile categories. One of the main contributions is to propose a point cloud sequence data set labeled point by point, and the category of the data set provides an unseen level of detail and categories of outliers. The outlier category is important for self-driving car LiDAR, because this category includes erroneous LiDAR measurements caused by light reflection or other reasons. Another contribution of SemanticKITTI is based on the contribution of the KITTI database, making SemanticKITTI the first serial tagging based on mobile cars, providing real world with many complex and changeable traffic conditions and environment data and benchmarks. The original KITTI data contain 22 sequences. After labelling all of them in SemanticKITTI, set Sequences 00-10 as training data, Sequences 11-21 as test data, and we set Sequence 08 from training data as validation dataset.

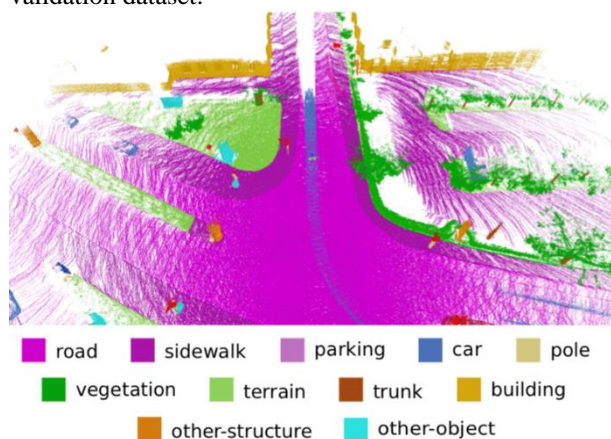


Figure 4.1 SemanticKITTI [24].


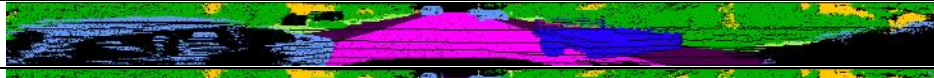

4.2 Evaluation

In Table 4.2.1, the measurement standard is IoU. Values in table are calculated as percentages, and we throw them into SemanticKITTI Competition in Codalab to compete with many Competitive competitor who use SemanticKITTI for semantic segmentation. It uses SemanticKITTI's seq 11-21 as the testing dataset for testing. In Table 4.2.1, the Salsanext [23] we reproduced was the most outstanding projection-based method 3D point cloud segmentation network that provided program reproduction at the time. But like the Lovasz softmax issue about batch size, if we only use a single 2080 Ti, the limited GPU Memory makes batch size limited to 4. Thus, if we directly reproduce Salsanext with batch size 4, our mIoU will drop to 57.2. Moreover, Table 4.2.1 expresses which categories of point cloud segmentation methods can be defeated based on SalsaNext. The more important contribution we want to mention is that, without changing all the parameters except batch size, we only add our CGAN adversarial training method to pay attention to the correlation between adjacent pixels, which can make Mean-IoU apparently increase. Furthermore, model parameters obtained by our training method in inference are the same as Salsanext.

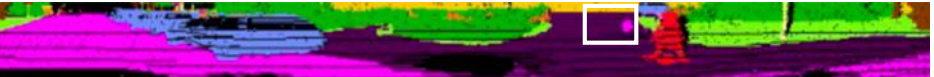
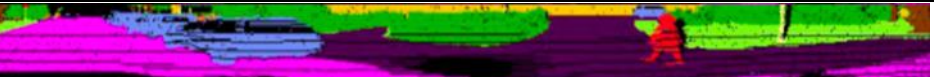
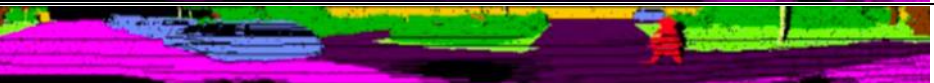
SalsaNext in Table 4.2.1 uses mIoU score proposed in their paper. Different results from our own training lies in the hardware equipment mentioned in their paper, that is, SalsaNext uses a huge GPU Memory with the loss function of Lovasz softmax to reach this number. In Table 4.2.1, our complete architecture will also add ASPP, SE Block, and change to RMI Loss. We also successfully surpassed Salsanext in the end, which uses better hardware equipment to train. Table 4.2.1 can see that proposed network can surpass powerful SalsaNext, even if it uses poorer hardware equipment than Salsanext. Moreover, Table 4.2.2 is visual comparison between the proposed network unplugged CGAN discriminator and complete proposed network with CGAN discriminator and Ground Truth.

	Approach	car	bicycle	motor-cycle	truck	other-vehicle	person	bicyclist	motor-cyclist	road	parking	sidewalk	Other-ground	building	fence	vegetation	trunk	terrain	Pole	Traffic-sign	mean-IOU
Point-based	Pointnet	46.3	1.3	0.3	0.1	0.8	0.2	0.2	0.0	61.6	15.8	35.7	1.4	41.4	12.9	31.0	4.6	17.6	2.4	3.7	14.4
	Pointnet++	53.7	1.9	0.2	0.9	0.2	0.9	1.0	0.0	72.0	18.7	41.8	5.6	62.3	16.9	46.5	13.8	30.0	6.0	8.9	20.1
	RandLA-Net	94.2	26.0	25.8	40.1	38.9	49.2	48.2	7.2	90.7	60.3	73.7	20.4	86.9	56.3	81.4	61.3	66.8	49.2	47.1	53.9
	LatticeNet	88.6	12.0	20.8	43.3	24.8	34.2	39.9	60.9	88.8	64.6	73.8	25.5	86.9	55.2	76.4	67.9	54.7	41.5	42.7	52.2
Projection-based	SqueezeSegV2	81.8	18.5	17.9	13.4	14.0	20.1	25.1	3.9	88.6	45.8	67.6	17.7	73.7	41.1	71.8	35.8	60.2	20.2	36.3	39.7
	SqueezeSegV2-CRF	82.7	21.0	22.6	14.5	15.9	20.2	24.3	2.9	88.5	42.4	65.5	18.7	73.8	41.0	68.5	36.9	58.9	12.9	41.0	39.6
	RangeNet++	91.4	25.7	34.4	25.7	23.0	38.3	38.8	4.8	91.8	65.0	75.2	27.8	87.4	58.6	80.5	55.1	64.6	47.9	55.9	52.2
	SqueezeSegV3	92.5	38.7	36.5	29.6	33.0	45.6	46.2	20.1	91.7	63.4	74.8	26.4	89.0	59.4	82.0	58.7	65.4	49.6	58.9	55.9
	SalsaNext (train by ourself)	91.1	43.7	34.3	37.5	29.9	59.1	53.6	30.4	90.9	60.4	73.2	23.6	87.6	56.6	79.5	58.6	63.9	51.3	62.0	57.2
	SalsaNext (+ CGAN)	92.0	47.4	39.0	25.6	34.6	59.0	57.6	27.8	91.8	64.9	75.0	21.3	88.8	60.5	81.3	64.7	64.7	54.6	61.2	58.5
	SalsaNext (+ CGAN, RMI)	90.8	47.6	40.9	43.2	29.7	59.6	58.4	28.9	92.0	63.7	76.1	26.6	89.0	61.2	82.7	63.4	67.5	55.2	60.9	58.9
	Our Adv. Net	90.1	50.5	44.3	35.7	33.0	58.7	52.5	21.6	91.7	64.8	76.5	27.5	90.0	63.5	82.9	64.7	67.5	55.7	60.7	59.6
	Salsanext (Paper declared)	91.9	48.3	38.6	38.9	31.9	60.2	59.0	19.4	91.7	63.7	75.8	29.1	90.2	64.2	81.8	63.6	66.5	54.3	62.1	59.5




Table 4.2.1 Quantitative comparison of various point cloud segmentation networks.

	Segmentation Net only
	With Adversarial Net
	Ground Truth

(a) There are related diagrams in the architecture Figure 3.2.1. The truck point cloud that should have been identified as dark blue, but some of it has been identified as light blue cars.

	Segmentation Net only
	With Adversarial Net
	Ground Truth

(b) The whole piece of point cloud need identified as purple, but there is a part of the road surface identified as pink.

	Segmentation Net Only
	With Adversarial Net
	Ground Truth

(c) In addition to the obvious yellow and orange categories on the left, entire yellow point cloud on the right that was originally identified as building, but a part of the red point cloud identified as people category.

Table 4.2.2 Three qualitative visual examples. Key part is framed by white frames in the pictures of Without Adversarial.

5. CONCLUSION

In this paper, the segmentation network is augmented with an adversarial network to improve the contextual continuity in adjacent regions of the predicted segmentation map. We use adversarial network to update

generator without additional post-processing like CRF or KNN. In generator, we also integrated ASPP module to increase the receptive field, and introduce data augmentation to increase the data variety. For future work, in order to train with a different raw point cloud dataset, we will introduce semi-supervised approach to

automate the initial the initial labeling task.

ACKNOWLEDGEMENT

This work is supported by Institute for Information Industry.

REFERENCES

- [1] I. Bogoslavskyi and C. Stachniss, "Efficient Online Segmentation for Sparse 3D Laser Scans," *Journal of Photogrammetry, Remote Sensing and Geoinformation Science*, 2017.
- [2] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *Proceedings of Conference on Computer Vision and Pattern Recognition*, Honolulu, Hawaii, 2017.
- [3] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud," *Proceedings of International Conference on Robotics and Automation*, Brisbane, Australia, 2018.
- [4] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "Squeezesegv2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud," *Proceedings of International Conference on Robotics and Automation*, Montreal, Canada, 2019.
- [5] C. Xu, B. Wu, Z. Wang, W. Zhan, P. Vajda, K. Keutzer, and M. Tomizuka, "Squeezesegv3: Spatially-Adaptive Convolution for Efficient Point Cloud Segmentation," *Proceedings of European Conference on Computer Vision*, Glasgow, UK (Virtual), 2020.
- [6] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, "Semantic Segmentation Using Adversarial Networks," *Proceedings of Workshop on Adversarial Training on Neural Information Processing Systems*, Barcelona, Spain, 2016.
- [7] N. Souly, C. Spampinato, and M. Shah, "Semi-Supervised Semantic Segmentation Using Generative Adversarial Network," *Proceedings of International Conference on Computer Vision*, Venice, Italy, 2017.
- [8] I. J. Goodfellow, J. P. Abadie, M. Mirza, B. Xu, D. W-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Networks," *Proceedings of Conference on Neural Information Processing Systems*, Montréal Canada, 2014.
- [9] M. Mirza, S. Osindero, "Conditional Generative Adversarial Nets," arXiv preprint arXiv:1411.1784, 2014.
- [10] P. Isola, J.-Y. Zhu, T. Zhou, A. A. Efros, "Image-to-Image Translation with Conditional Adversarial Networks," *Proceedings of Conference on Computer Vision and Pattern Recognition*, Honolulu, Hawaii, 2017.
- [11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, Nevada, 2016.
- [12] F. Yu and V. Koltun, "Multi-Scale Context Aggregation by Dilated Convolutions," *Proceedings of International Conference on Learning Representations*, San Juan, Puerto Rico, 2016.
- [13] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.
- [14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Robinovich, "Going Deeper with Convolutions," *Proceedings of Conference on Computer Vision and Pattern Recognition*, Boston, Massachusetts, 2015.
- [15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, Nevada, pp.2818-2826, 2016.
- [16] S. Xie, R. Girshick, and P. Dollár, "Aggregated Residual Transformations for Deep Neural Networks," *Proceedings of Conference on Computer Vision and Pattern Recognition*, Honolulu, Hawaii, 2017.
- [17] J. Hu, L. Shen, and G. Sun, "Squeeze-and-Excitation Networks," *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, Salt Lake City, Utah, 2018.
- [18] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network," *Proceedings of Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, 2016.
- [19] P. Wang, P. Chen, Y. Yuan, D. Liu, Z. Huang, X. Hou, and G. Cottrell, "Understanding Convolution for Semantic Segmentation," *Proceedings of Winter Conference on Applications of Computer Vision*, Lake Tahoe, NV, USA, 2018.
- [20] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs," *Proceedings of Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, United States, 2017.
- [21] S. Zhao, Y. Wang, Z. Yang, and D. Cai, "Region Mutual Information Loss for Semantic Segmentation," *Proceedings of International Conference on Neural Information Processing Systems (NeurIPS)*, Canada Vancouver, 2019.
- [22] E. E. Aksoy, S. Baci, and S. Cavdar, "SalsaNet: Fast Road and Vehicle Segmentation in LiDAR Point Clouds for Autonomous Driving," *Proceeding of IEEE Intelligent Vehicles Symposium*, France, 2019.
- [23] T. Cortinhal, G. Tzelepis, and E. E. Aksoy, "SalsaNext: Fast, Uncertainty-Aware Semantic Segmentation of LiDAR Point Clouds for Autonomous Driving," *Proceedings of International Symposium on Visual Computing*, San Diego, CA, 2020.
- [24] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences," *Proceedings of International Conference on Computer Vision*, Seoul, Korea, 2019.
- [25] M. Berman, A. R. Triki, and M. B. Blaschko, "The Lovász-Softmax Loss: A Tractable Surrogate for the Optimization of the Intersection-Over-Union Measure in Neural Networks," *Proceedings of Conference on Computer Vision and Pattern Recognition*, Salt Lake City, Utah, 2018.
- [26] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," *Proceeding of Journal of Robotics Research*, 2013.
- [27] J. L. Liou, A. Tsai, C. S. Fuh, and F. Huang, "MamboNet: Adversarial Semantic Segmentation for Autonomous

Driving,” *Proceedings of International Symposium on Geometry and Vision*, Auckland, New Zealand, 2021.

- [28] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, “RangeNet++: Fast and Accurate LiDAR Semantic Segmentation,” *Proceedings of International Conference on Intelligent Robots and Systems*, Macau, China, 2019.
- [29] A. Tsai, F. Huang, J. L. Liou, M. C. Tseng, and P. S. Hu, “Foreign Objects Removal for Self-driving Mapping,” *Proceeding of IEEE International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan)*, Taoyuan, Taiwan, 2020.