EasyChair Preprint

№ 7026

1D Self-Attention Network for Point Cloud Semantic Segmentation Using Omnidirectional LiDAR

Takahiro Suzuki, Tsubasa Hirakawa, Takayoshi Yamashita and Hironobu Fujiyoshi

November 10, 2021

1D Self-Attention Network for Point Cloud Semantic Segmentation using Omnidirectional LiDAR

Takahiro Suzuki[1], Tsubasa Hirakawa[1], Takayoshi Yamashita[1], and Hironobu Fujiyoshi[1]

Chubu University, 1200 Matsumotocho, Kasugai, Aichi, Japan
{stkin26@mprg.cs , hirakawa@mprg.cs , yamashita@isc , fujiyoshi@isc}.chubu.ac.jp

Abstract. Understanding the environment around a vehicle is essential for automated driving technology. For this purpose, omnidirectional LiDAR is used for obtaining surrounding information, and point cloud-based semantic segmentation methods have been proposed. However, these methods require time to acquire point cloud data and to process the point cloud, which causes a significant positional shift of objects in practical application scenarios. In this paper, we propose a 1D self-attention network (1D-SAN) for LiDAR-based point-cloud semantic segmentation, which is based on a 1D-CNN for real-time pedestrian detection of omnidirectional LiDAR data. Because the proposed method can sequentially process segmentation during data acquisition with omnidirectional LiDAR, we can reduce the processing time and suppress positional shift. Moreover, for improving segmentation accuracy, we use the intensity as input data and introduce a self-attention mechanism into the method. The intensity enables us to consider object texture. The self-attention mechanism can consider the relationship between point clouds. Experimental results with the SemanticKITTI dataset show that the intensity input and the self-attention mechanism in the proposed method improve accuracy. In particular, the mechanism contributes to improving the accuracy for small objects. Also, we show that the processing time of the proposed method is faster than the other point-cloud segmentation methods.

Keywords: Point Cloud · Semantic Segmentation · Self-Attention.

1 Introduction

With automated driving technology, it is essential to understand the environment around the vehicle. For this reason, research on automatic driving has attracted a great deal of attention, and typical functions of driving support systems include detecting objects in the vicinity of the vehicle and predicting the path of pedestrians. In particular, object detection is a fundamental method for automated driving. It can be categorized into two approaches based on the input

data: RGB images from onboard cameras and 3D point clouds acquired by Light Detection and Ranging (LiDAR) [14, 15, 3, 21].

LiDAR obtains 3D information and intensity by measuring the time it takes for an infrared laser beam to reflect off an object and return. Among the several types of LiDAR devices, omnidirectional LiDAR acquires 3D information in 360 degrees in all directions with the LiDAR as the origin by irradiating the laser toward the surrounding area while rotating.

Many semantic segmentation methods using omnidirectional LiDAR have been proposed [14, 15, 3, 21, 17, 13, 16]. However, the acquisition of omnidirectional LiDAR data requires a constant amount of time. In addition to the data acquisition time, considering the processing time for point clouds, the time is long. This causes a significant positional shift of objects and is a crucial problem for the practical automated driving scenario.

In this paper, we propose a 1-dimensional self-attention network (1D-SAN), a semantic segmentation method for omnidirectional LiDAR-based point clouds. The key idea of the proposed method is processing a part of point clouds in entire 360-degree point cloud data sequentially. The method is based on the 1-dimensional convolutional neural network (1D-CNN) [8], which is a pedestrian detection method for omnidirectional LiDAR. The 1D-CNN regards the distance values obtained from a LiDAR as 1-dimensional waveform data and uses the data for network input. This enables us to sequentially process omnidirectional LiDAR data during data acquisition, which can suppress the positional shift of objects. We extend the 1D-CNN for a semantic segmentation task to deal with multiple object classes. Moreover, we propose using reflection intensity values for network input and introduce a self-attention mechanism. Since the reflection intensity differs depending on the material of an object, we can consider the texture of objects. As the self-attention mechanism for 1D waveform data, we propose the 1D self-attention block (1D-SAB), which is based on self-attention block [25]. By introducing 1D-SAB, we can consider the relationship between point clouds. Due to the reflection intensity and 1D-SAB, we can improve the segmentation accuracy while maintaining the sequential process of omnidirectional LiDAR data. Experimental results obtained with the SemanticKITTI dataset show that the intensity input and the self-attention mechanism in the proposed method improve accuracy. In particular, the mechanism contributes to improving the accuracy for small objects. Also, we show that the processing time of the proposed method is faster than the other point cloud segmentation methods.

The contributions of this paper are as follows:

- This paper proposes a point cloud-based semantic segmentation method.
- The proposed method can process segmentation sequentially while acquiring data. Therefore, our method is faster than existing semantic segmentation methods, and we can reduce positional shift.
- The proposed method improves the accuracy of semantic segmentation, especially for small objects, by self-attention.

2 Related Work

Many methods using point clouds have been proposed [14, 15, 3, 21, 17, 13, 16, 8, 10, 26, 1, 11, 4, 9, 22, 23, 12, 20, 7, 18, 24, 19]. These methods can be categorized into three approaches. The first approach treats the point cloud as voxels [10, 26]. The second approach projects the point cloud onto an image and treats it as an image [3, 21, 1, 4, 22, 23, 12]. The third approach uses a 3D point cloud [17, 16, 1, 11, 9, 20, 7, 18, 24, 19]. In addition to the above approaches, there is pedestrian detection using a 1D-CNN. It identifies whether the area is a pedestrian area or not sequentially from the process of acquiring data from omnidirectional LiDAR [8]. In this section, we briefly introduce common methods in the point cloud approach.

2.1 Voxel-based Method

The voxel-based method first converts a 3D point cloud as a voxel representation. Then, the voxelized point cloud data is input to a network consisting of 3D convolutions to obtain results [26].

In VoxelNet, the 3D point cloud is divided into voxels, and a network for object detection is proposed [26]. VoxelNet is composed of a feature learning network (FLN), convolutional middle layers, and region proposal network (RPN). First, the 3D information is divided into equally spaced voxels by FLN, and the shape information in each voxel is obtained. At this time, the feature values of each point in the voxel are also calculated and combined with the feature values of each voxel. Next, 3D convolutional processing is performed using convolutional middle layers to aggregate the features into voxel units. Finally, object regions are detected by RPN.

The voxel-based method makes it easy to retain the original information of a 3D point cloud, and smooth feature extraction by 3D convolution is possible. It also improves on the sparseness of 3D point clouds by grouping them by voxel, making them easier to handle for each task. However, due to the cubical representation of voxel data, this is computationally expensive and decreases the process speed.

2.2 Image-based Method

In the image-based method, the 3D point cloud data is first projected onto a 2D image. The projected 2D image is then subjected to 2D convolutions in the same way as normal images [21].

For SqueezeSeg [21], a network was proposed for projecting point cloud data acquired from LiDAR onto a cylinder and treating and processing it as an image. By using SqueezeNet to extract features, we have been able to speed up processing while maintaining the original system. SqueezeNet reduces the number of parameters by introducing a Fire Module into the neural network, and it preserves accuracy by performing downsampling backward. SqueezeSeg introduces Fire Deconv to this SqueezeNet and upsamples feature maps. Fire Deconv

allows for even faster processing. In addition, the accuracy of segmentation is improved by modifying labels with a recurrent conditional random field. In summary, SqueezeSeg achieves faster processing while maintaining accuracy with fewer parameters.

By transforming a 3D point cloud into a 2D image, we can apply a 2D convolutional process and increase the processing speed. However, there is a possibility that the original information of the 3D point cloud is missing, for example, a pixel and its neighboring pixels are not the actual neighboring points in a transformed image.

2.3 3D Point Cloud-based Method

In the 3D point cloud-based method, a point cloud is directly input to a network for processing [14, 15]. We input (x, y, z) coordinate information and the reflection intensity values of point clouds into a network.

For PointNet [14], a network was proposed that can be applied to tasks such as three-class classification and segmentation. PointNet is composed of a spatial transformer network (STN), a classification network, and a segmentation network. First, we reduce the noise for the input point cloud in STN. The next step is to extract the features of each point cloud from the convolution process by using a classification network. After that, max pooling is used to extract the overall features and classify them. In the case of segmentation, the overall features extracted by the classification network and the local features of each point cloud are combined and input to the segmentation network. The convolution process is performed several times again, and segmentation is performed for each point cloud.

PointNet may lack detailed spatial information, and as a result, may not be able to capture the local structure. For this problem, there is PointNet++ [15], which is an improved version of PointNet with higher accuracy. In PointNet++, a network that can capture local features was proposed that applies PointNet hierarchically. In PointNet++, PointNet is used for local feature extraction. It is also possible to extract pseudo local features by inputting neighboring points that have been clustered. This solves the problems of PointNet and improves the accuracy of class classification and segmentation.

Thus, the original information of the 3D point cloud is retained, and accurate feature extraction is possible. These methods also eliminates the computational cost of converting to voxels, etc. However, processing 3D point clouds as they are requires a huge amount of storage space. The associated computational cost of processing a point cloud is also high, which may result in a reduction in processing speed.

2.4 1D-CNN for Pedestrian Detection

One of the problems with the above mentioned approaches is that the detection process takes a long time. This causes a gap between the detected position and

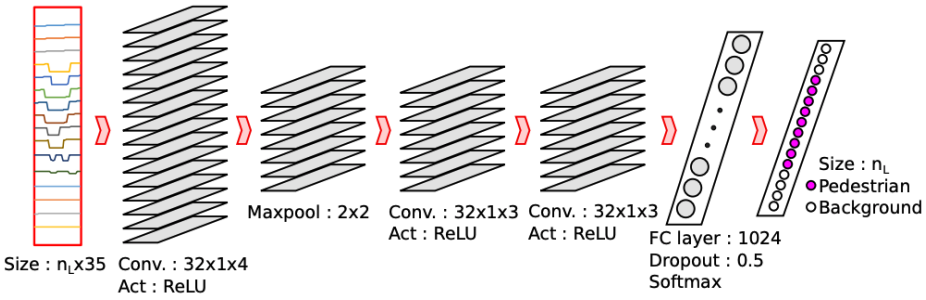


Fig. 1. Network structure of 1D-CNN. Quote from [8].

the actual position when a LiDAR device is mounted on a fast-moving object such as an automobile.

To overcome this problem, the 1D-CNN [8] was proposed. It can suppress positional shift in the pedestrian detection problem. For pedestrian detection using the 1D-CNN, the distance values obtained from LiDAR are regarded as 1D waveform data for each laser ID, and they are sequentially input to the 1D-CNN to enable pedestrian detection along with LiDAR rotation. The structure of the 1D-CNN is shown in Fig. 1. The network consists of three convolutional layers and one fully-connected layer. The input data size is $n_L \times 35$, where n_L is the number of laser IDs of LiDAR. Since the average width of a pedestrian is about 7 degrees horizontally, the number of point clouds for 7 degrees is 35, which is the width. During the actual driving of the vehicle, when the number of point clouds for pedestrians is secured with the rotation of the LiDAR, the point clouds are input to the network sequentially to identify whether they are pedestrians or background. Furthermore, clustering is applied to the point clouds identified as pedestrians to achieve higher accuracy.

This sequential processing along with the rotation of the LiDAR makes it possible to minimize the gap between the detected position of a pedestrian and the actual position even when driving. However, in this method, only pedestrians are detected. In actual automated driving, it is also important to detect objects other than pedestrians. In this paper, therefore, we propose a semantic segmentation method for omnidirectional LiDAR based on the 1D-CNN approach. In addition to the distance value, we add reflection intensity as an input and introduce a self-attention mechanism. Because these enable us to consider the texture of each object and the relationship between neighboring point clouds, we can improve the segmentation accuracy while keeping the computational cost reasonable.

3 Proposed Method

In this section, we introduce the details of the proposed semantic segmentation method. Figure 2 shows the network structure of our proposed 1-dimensional

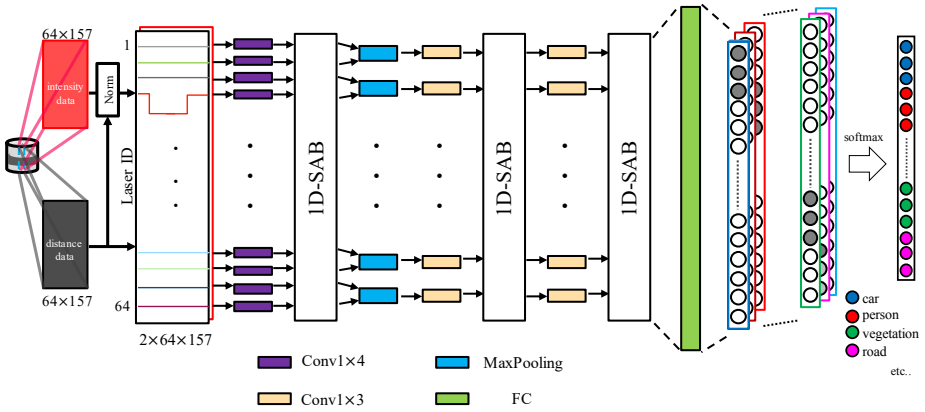


Fig. 2. Network structure of 1D-SAN.

self-attention network (1D-SAN). We first obtain distance and reflection intensity values from LiDAR and interpolate those missing values. Then, we assume these LiDAR data to be 1-dimensional waveform data and concatenate them for channel direction. In addition, we input the processed LiDAR data to the 1D-SAN. The 1D-SAN outputs the class probabilities for each piece of waveform data. This process enables us to achieve multi-class classification while maintaining the advantages of the 1D-CNN. Moreover, the proposed network contains 1-dimensional self-attention blocks (1D-SABs). The 1D-SAB utilizes the relative position between point clouds for the weighting process, so we can consider the important relationships between point clouds.

3.1 Network Structure

Here, we describe the network structure of 1D-SAN in detail. As shown in Fig. 2, 1D-SAN consists of three convolutional layers, three 1D-SAB layers, and one fully-connected layer. In all convolutional layers, convolution is performed only in the horizontal direction. The longitudinal size of the input data and the number of output units depend on the number of lasers in the LiDAR. The LiDAR used in this study is 64 because it irradiates 64 lasers. The lateral size is set to 157 points per 30 degrees, which is optimal for semantic segmentation. The number of input channels is two: distance value and intensity. 1D convolution of the input data is performed for each laser ID. After each convolutional layer, the data is input to 1D-SAB to take into account the relationship between the point clouds. The features are then combined in the fully-connected layer, and the softmax function calculates the probability of each class for each laser ID in the center of the input data, and it outputs the identification results. By doing this sequentially, we can achieve semantic segmentation for all directions.

3.2 Preprocessing Point Cloud Data

Point cloud data is a set of points where the laser beam from the omnidirectional LiDAR is reflected back to the object, and the distance and intensity values can be obtained from the time. However, for omnidirectional LiDAR, there are scenes where it is difficult to acquire reflected light, such as the sky and specular objects. If the reflected light cannot be obtained, the value at that point cannot be obtained, and the value becomes an outlier, so interpolation of the value at the outlier part is necessary.

In this study, the outliers of the distance values are interpolated with the maximum distance value of 120 assuming that the object is empty when the irradiation angle is 0 degrees or more and with the corresponding value assuming that the laser hits the ground when the irradiation angle is less than 0 degree. Similarly, the outliers of the intensity are interpolated with 0.0 when the irradiation angle is more than 0 degrees and with the average intensity of the ground class such as roads and sidewalks, 0.29, when the irradiation angle is less than 0 degrees. The interpolated distance values and intensities are combined for each laser ID to create waveform data for each laser ID.

3.3 Normalization of Intensity Value

To improve the discrimination accuracy of semantic segmentation, we add the intensity of objects to the input data. The intensity is weakened on highly reflective objects such as metal because the light is diffused, and it is strengthened on less reflective objects such as cloth because the light is returned exactly. Therefore, by adding the intensity as an input to the network, we can expect to identify objects on the basis of their texture. The texture of objects refer to the color and material information of the object in here. The intensity decreases as the distance to an object increases, and the value returned becomes smaller.

We correct the value by normalizing it with the distance value from LiDAR to suppress the effect of attenuation due to distance. To normalize the intensity, the law of light decay is used. The law of light attenuation states that the intensity of light is inversely proportional to the square of the distance from the light source to the object [6]. We define the normalization of the intensity by using the distance value as follows:

$$I' = I \times (2 \times d)^2 \tag{1}$$

where I' is the intensity after normalization, I is the intensity before normalization, and d is the distance value obtained from LiDAR. With Eq. (1), it is possible to recover the value attenuated by the distance and use the intensity as input.

3.4 1-Dimensional Self-Attention Block (1D-SAB)

Here, we give details on 1D-SAB. Figure 3 shows the detailed structure of 1D-SAB. The created 1D waveform data is input into the 1D-SAB for each laser

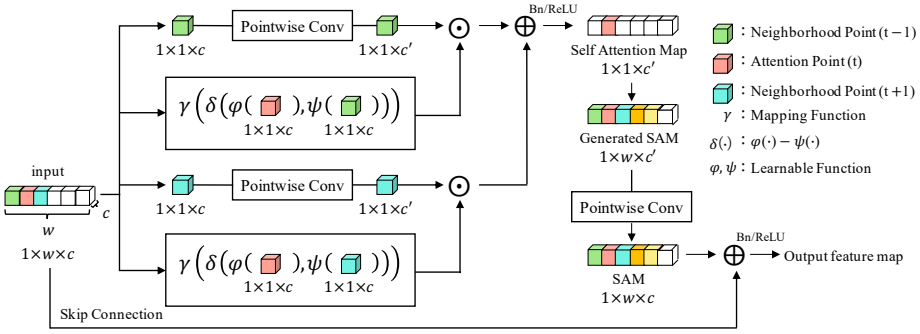


Fig. 3. Detailed structure of 1D-SAB.

ID. The input data is processed one point at a time, and the self-attention of the corresponding point is calculated. When the red value in Fig. 3 is the point of interest for the process, the green is neighborhood 1, and the blue is neighborhood 2. For each neighboring point, we apply pointwise convolution.

We input the points of interest and neighboring points into the learnable functions $\varphi(\cdot)$ and $\psi(\cdot)$. Then, $\varphi(\cdot)$ and $\psi(\cdot)$ are used for the relational function δ, which is defined as

$$\delta\left(\varphi(\cdot), \psi(\cdot)\right) = \varphi(\cdot) - \psi(\cdot). \tag{2}$$

Then, the mapping function $\gamma(\cdot)$ aligns the number of channels with the output of the first process. Then, we calculate the element-wise product with the above mentioned features by pointwise convolution.

The self-attention map (SAM) is generated by performing this process for neighboring points and summing them up. The generated SAMs are made to have the same number of channels as the input channels by pointwise convolution. To this output, the input data is added as a skip mechanism to form the final output. By using 1D-SAB, we can give a large weight to the important positions among the point clouds and consider the relationship among the point clouds.

4 Experiments

In this section, we evaluate the proposed method. In our experiments, we verify the effectiveness of the reflection strength and 1D-SAB for semantic segmentation. We also compare the accuracy with other methods.

4.1 Summary of Experiment

Dataset We used SemanticKITTI [2] for our evaluation. SemanticKITTI is a real-world dataset that was created on the basis of the KITTI dataset [5] for autonomous driving purposes.

The 3D point cloud data was acquired by HDL-64E. The dataset consisted of 22 scenes, and the number of frames was 43,000. Among them, 21,000 frames taken in scenes 00 to 10 were used for training, and 22,000 frames from scenes 11 to 21 were used for testing. In the training set, we used 4,080 frames of scene 08 for validation and the other frames for training.

SemanticKITTI annotates all the point clouds in the KITTI dataset and defines 22 classes, including people, cars, and buildings. People and cars are also classified by the presence or absence of motion. In this study, because dynamic objects are treated in the same way as static objects, we used 19 classes in our experiments.

Methods for comparison As methods for comparison, we used the following methods. To evaluate the effectiveness of the reflection intensity and 1D-SAB and the computational time, we compared the performances of the following methods.

- 1D-CNN: We extended the output of the conventional 1D-CNN [8] for predicting the probability of 19 classes. Note that it did not use the reflection intensity as an input.
- Ours (w/o SAB): We used the reflection intensity as an input and removed 1D-SAB from the proposed method.
- Ours (w/ SAB): We used both the reflection intensity input and 1D-SAB.

To train these methods, we set the number of training epochs to 20 and the batch size to 24. Cross entropy loss was used as the loss function, MomentumSGD was used as the optimization method, and the initial learning rate was 0.01. During training, the learning rate was decreased by a factor of 1/2 per epoch.

Moreover, by using the test set of SemanticKITTI dataset, we compared the segmentation performance with four point-cloud segmentation methods: PointNet [14], PointNet++ [15], SPGraph [9], and SPLATNet [18].

Evaluation metrics We used intersection over union (IoU) as an evaluation metric. IoU measures how well the segmentation result matches the correct label for each point cloud. If the number of point clouds that answered correctly is true positive (TP), the number of point clouds that predicted the correct class as another class is false positive (FP), and the number of point clouds that predicted another class as the correct class is false negative (FN), the IoU can be defined as:

$$\text{IoU} = \frac{TP}{TP + FP + FN}. \tag{3}$$

The mean IoU is used as an overall evaluation metric. The mean IoU is obtained by taking the average of the IoU for each class, and it is defined by

$$\text{mIoU} = \frac{1}{C} \sum_{i=1}^{C} \frac{TP}{TP + FP + FN}, \tag{4}$$

where C indicates the number of classes.

Table 1. Evaluation of effectiveness of proposed method on SemanticKITTI test set (Sequences 11 to 21). IoU scores are given in percentage (%).

Approach	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic sign	mean-IoU
1D-CNN [8]	54.7	0.0	0.0	0.0	0.3	0.0	0.0	0.0	69.6	3.9	39.3	0.0	59.9	20.6	57.9	4.2	40.2	15.0	0.0	19.4
Ours (w/o SAB)	57.1	0.0	0.0	0.3	0.1	0.0	0.0	0.0	**72.6**	10.7	**43.0**	0.0	61.3	17.6	59.9	3.7	42.0	13.9	4.6	20.2
Ours (w SAB)	**58.1**	**3.3**	**2.2**	**4.8**	**3.4**	**6.0**	**10.7**	**0.4**	**72.6**	**16.8**	38.7	1.2	**66.9**	**28.4**	**70.7**	**5.5**	**54.7**	**29.7**	**32.2**	**26.6**

Table 2. Quantitative comparison with other segmentation methods on SemanticKITTI test set (Sequences 11 to 21). IoU scores are given in percentage (%).

Approach	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic sign	mean-IoU
PointNet [14]	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	17.7	2.4	3.7	14.6
PointNet++ [15]	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
SPGraph [9]	**68.3**	0.9	**4.5**	0.9	0.8	1.0	6.0	0.0	49.5	1.7	24.2	0.3	68.2	22.5	59.2	27.2	17.0	18.3	10.5	20.0
SPLATNet [18]	66.6	0.0	0.0	0.0	0.0	0.0	0.0	0.0	70.4	0.8	41.5	0.0	**68.7**	27.8	**72.3**	**35.9**	35.8	13.8	0.0	22.8
Ours (w/o SAB)	57.1	0.0	0.0	0.3	0.1	0.0	0.0	0.0	**72.6**	10.7	**43.0**	0.0	61.3	17.6	59.9	3.7	42.0	13.9	2.4	20.2
Ours (w SAB)	58.1	**3.3**	2.2	**4.8**	**3.4**	**6.0**	**10.7**	**0.4**	**72.6**	16.8	38.7	1.2	66.9	**28.4**	70.7	5.5	**54.7**	**29.7**	**32.2**	**26.6**

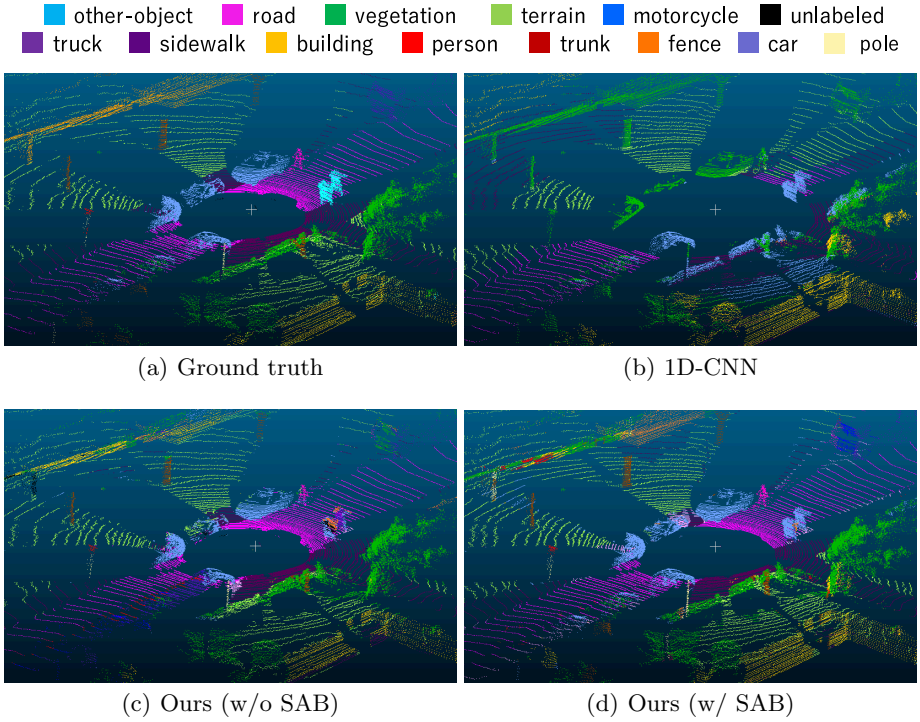
4.2 Evaluating Effectiveness of Intensity

First, we evaluated the effectiveness of intensity and 1D-SAB. Table 1 shows the accuracy comparison between the conventional and the proposed methods on the test set of SemanticKITTI.

Comparing 1D-CNN and Ours (w/o SAB), we can see that the mean-IoU was improved by 0.8 pts. by introducing the reflection intensity. Moreover, we focused on the IoU of each class. By introducing the reflection intensity, the IoU of 9 out of 19 classes was improved, while the IoUs of some classes, e.g., fence and trunk, were decreased even when the intensity was taken into account. This suggests that, depending on the object, the accuracy may not be improved by considering the intensity. However, it was confirmed that the IoU for car, road, and traffic-sign, which are considered to be important in automated driving, were improved. For the classes with a decreasing IoU, the accuracy was almost equal to that of the conventional method. These results show that the introduction of intensity is effective in semantic segmentation.

4.3 Evaluating Effectiveness of 1D-SAB

Next, we evaluated the effectiveness of 1D-SAB. From Table 1, the mean-IoU of Ours (w/ SAB) was 26.6%, which was 6.4 pts. higher than that of Ours (w/o SAB). Therefore, it can be said that considering the relationship between point clouds with a self-attention mechanism is effective for improving the accuracy of



(a) Ground truth (b) 1D-CNN

(c) Ours (w/o SAB) (d) Ours (w/ SAB)

Fig. 4. Visualization results on SemanticKITTI test set.

semantic segmentation. In terms of the IoU for each class, the introduction of 1D-SAB resulted in the highest IoU for 17 of the 19 classes. In particular, the accuracy for small objects such as bicyclist, pole, and traffic-sign was greatly improved. Therefore, we can say that the use of 1D-SAB, which determines weights, contributes to the improvement of accuracy, especially for classes with a small number of point clouds such as small objects.

4.4 Qualitative Evaluation

Here, we qualitatively evaluated the segmentation results. Figure 4 shows visualization results for each method. As shown in Fig. 4(b), the 1D-CNN recognized car as vegetation and road as terrain in some parts. Ours (w/o SAB) was able to recognize cars and roads better than 1D-CNN. Moreover, Ours (w/ SAB) was able to recognize objects closer to the ground truth, and the recognition accuracy for small objects such as pole and trunk was also improved.

4.5 Comparison of Accuracy with Other Methods

Here, we compare the performance with the other segmentation methods with the test set of the SemanticKITTI dataset. A comparison of the accuracy with

Table 3. Comparison of processing speed

	PointNet [14]	1D-CNN [8]	Ours (w/o SAB)	Ours (w SAB)
Speed(msec)	208.333	0.0814	0.0888	0.3488

the other methods is shown in Tab. 2. The proposed method with 1D-SAB had the highest accuracy with a mean-IoU of 26.6%. Focusing on the IoU for each class, the proposed method with 1D-SAB had the highest accuracy in 11 out of 19 classes. In particular, the accuracy for small objects such as fence, pole, and traffic-sign was greatly improved compared with the other methods. It can also be seen that the remaining eight classes had almost the same level of accuracy as the other methods. However, even without 1D-SAB, the mIoU was improved except for SPLATNet.

4.6 Comparison of Processing Speed

Finally, we show a comparison of the processing speeds. The speed of PointNet was calculated as the processing time for one rotation of LiDAR data, while the speed of 1D-CNN and the proposed method was calculated as the processing time for a range of data to be processed sequentially. All processing speed measurements were performed using an NVIDIA Quadro RTX 8000.

Table 3 shows that 1D-CNN was the fastest with a frequency of 0.0814 msec. Because 1D-CNN uses only distance values, and the network structure is also the smallest, it achieved a faster processing time. The model using the intensity as input data, i.e., Ours (w/o SAB), decreased the speed by 0.0074 msec compared with 1D-CNN. The model with 1D-SAB, which had the best accuracy comparison results, significantly decreased in speed to 0.3488 msec compared with 1D-CNN and Ours (w/o SAB). Since the omnidirectional LiDAR operates at 5 Hz to 20 Hz, the proposed method, which can process at a speed of less than 200 msec, can maintain real-time performance.

5 Conclusion

In this paper, we proposed the 1-dimensional self-attention network (1D-SAN) for omnidirectional LiDAR-based point-cloud semantic segmentation. The proposed method processes part of LiDAR data and estimates the semantic segmentation results sequentially, which can reduce the processing time. It uses intensity values as input and introduces a self-attention mechanism called 1D-SAB. The experimental results with the SemanticKITTI dataset showed that the use of the intensity value and 1D-SAB improved the accuracy of semantic segmentation while maintaining a lower computational time. In particular, 1D-SAB improved the accuracy for small objects.

Our future work includes achieving lower computational costs while maintaining a higher accuracy for practical automated driving applications.

References

1. Alonso, I., Riazuelo, L., Montesano, L., Murillo, A.C.: 3d-mininet: Learning a 2d representation from point clouds for fast and efficient 3d lidar semantic segmentation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE (2020)
2. Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., Gall, J.: SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In: Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV) (2019)
3. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3d object detection network for autonomous driving. In: IEEE Conference on Computer Vision and Pattern Recognition. p. 3 (2017)
4. Cortinhal, T., Tzelepis, G., Aksoy, E.E.: Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds for autonomous driving (2020)
5. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR). pp. 3354–3361 (2012)
6. Hasegawa, T., Tomizawa, R., Yamauchi, Y., Yamashita, T., Fujiyoshi, H.: Guided Filtering Using Reflected IR Image for Improving Quality of Depth Image. In: Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications. vol. 3, pp. 33–39 (2016)
7. Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A.: Randla-net: Efficient semantic segmentation of large-scale point clouds. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2020)
8. Kunisada, Y., Yamashita, T., Fujiyoshi, H.: Pedestrian-Detection Method Based on 1D-CNN During LiDAR Rotation. In: The 21st IEEE International Conference on Intelligent Transportation Systems (ITSC) (2018)
9. Landrieu, L., Simonovsky, M.: Large-scale point cloud semantic segmentation with superpoint graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
10. Maturana, D., Scherer, S.: Voxnet: A 3d convolutional neural network for real-time object recognition. In: Ieee/rsj International Conference on Intelligent Robots and Systems. pp. 922–928 (2015)
11. Meyer, G.P., Laddha, A., Kee, E., Vallespi-Gonzalez, C., Wellington, C.K.: Lasernet: An efficient probabilistic 3d object detector for autonomous driving. In: CVPR. pp. 12677–12686. Computer Vision Foundation / IEEE (2019)
12. Milioto, A., Vizzo, I., Behley, J., Stachniss, C.: RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. In: IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS) (2019)
13. Qi, C.R., Liu, W., Wu, C., Su, H., Guibas, L.J.: Frustum pointnets for 3d object detection from rgb-d data. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 918–927 (2018)
14. Qi, C.R., Su, H., Kaichun, M., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 77–85 (2017)
15. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems. pp. 5099–5108 (2017)

16. Simon, M., Milz, S., Amende, K., Gross, H.M.: Complex-yolo: Real-time 3d object detection on point clouds. arXiv preprint arXiv:1803.06199 (2018)
17. Spinello, L., Luber, M., Arras, K.O.: Tracking people in 3d using a bottom-up top-down detector. In: IEEE Robotics and Automation Society. pp. 1304–1310 (2011)
18. Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.H., Kautz, J.: SPLATNet: Sparse lattice networks for point cloud processing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2530–2539 (2018)
19. Tatarchenko*, M., Park*, J., Koltun, V., Zhou., Q.Y.: Tangent convolutions for dense prediction in 3D. CVPR (2018)
20. Thomas, H., Qi, C.R., Deschaud, J.E., Marcotegui, B., Goulette, F., Guibas, L.J.: Kpconv: Flexible and deformable convolution for point clouds. Proceedings of the IEEE International Conference on Computer Vision (2019)
21. Wu, B., Wan, A., Yue, X., Keutzer, K.: Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. ICRA (2018)
22. Wu, B., Zhou, X., Zhao, S., Yue, X., Keutzer, K.: Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In: ICRA (2019)
23. Xu, C., Wu, B., Wang, Z., Zhan, W., Vajda, P., Keutzer, K., Tomizuka, M.: Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. arXiv preprint arXiv:2004.01803 (2020)
24. Zhang, Y., Zhou, Z., David, P., Yue, X., Xi, Z., Gong, B., Foroosh, H.: Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (June 2020)
25. Zhao, H., Jia, J., Koltun, V.: Exploring self-attention for image recognition. In: CVPR (2020)
26. Zhou, Y., Tuzel, O.: Voxelnet: End-to-end learning for point cloud based 3d object detection. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 4490–4499 (2018)