Multi-Scale Voxel Class Balanced ASPP for LIDAR Pointcloud Semantic Segmentation

K S Chidanand Kumar
Great Wall Motors
Bangalore, Karnataka, INDIA
chidanand.kumar@gwmidc.in

Samir Al-stouhi
American Haval Motor Technology
Michigan, United States
samir.alstouhi@havalus.com

Abstract

This paper explores efficient techniques to improve PolarNet model performance to address the real-time semantic segmentation of LiDAR point clouds. The core framework consists of an encoder network, Atrous spatial pyramid pooling(ASPP)/Dense Atrous spatial pyramid pooling(DenseASPP) followed by a decoder network. Encoder extracts multi-scale voxel information in a top-down manner while decoder fuses multiple feature maps from various scales in a bottom-up manner. In between encoder and decoder block, an ASPP/DenseASPP block is inserted to enlarge receptive fields in a very dense manner. In contrast to PolarNet model, we use weighted cross entropy in conjunction with Lovasz-softmax loss to improve segmentation accuracy. Also this paper accelerates training mechanism of PolarNet model by incorporating learning-rate schedulers in conjunction with Adam optimizer for faster convergence with fewer epochs without degrading accuracy. Extensive experiments conducted on challenging SemanticKITTI dataset shows that our high-resolution-grid model obtains competitive state-of-art result of 60.6 mIOU @21fps whereas our low-resolution-grid model obtains 54.01 mIOU @35fps thereby balancing accuracy/speed trade-off.

1. Introduction

Scene understanding is an essential prerequisite for autonomous vehicles. Semantic segmentation helps in understanding the scene by predicting a meaningful class label for each individual sensor data point. Recent approaches to semantic segmentation exploit different data sources. Camera based approaches utilize either monocular [1] or stereo images [2], fish-eye cameras or depth cameras. These camera-based approaches have drawbacks such as limited field-of-view, difficult to operate under low-contrast conditions and inability to determine precise distances within the surrounding outdoor environment. On the other hand, LiDAR sensors, which use reflected laser pulses to scan the area around a vehicle, can overcome such limitations. LiDAR data is used to create a 360° point cloud, which solves the limited field of view problem experienced in camera-based systems and LiDAR data is more robust to changes in weather/ illumination issues in indoor/outdoor environments. Thus, they are generally considered as more important sensors than cameras for autonomous vehicles driving safety and are adopted by nearly all auto-makers today [3].

Even though LiDAR point clouds are superior compared to images, it has its own drawbacks like point clouds are sparse with varying density, highly unordered, noisy, lack colour and texture features that characterize the object classes. Such complexity, in addition to the dynamic nature of the environment, motivates us to improve performance of semantic-segmentation of LiDAR point clouds.

Current LiDAR based semantic segmentation can be categorized in three approaches. The first approach uses purely point-wise methods acting directly on the 3D point cloud [4, 5]. The second approach builds on top of the well developed field of image segmentation, which focuses on CNN architectures for segmenting RGB images [6, 7, 8, 9]. To this end, individual LiDAR sweeps are projected to 2D range images, which then serve as input to custom CNNs [10, 11, 12]. The resulting 2D predictions are then post-processed with non-learned Conditional Random Fields(CRF) or KNN-based voting to recover more accurate labels for each 3D point. The third approach partitions the point cloud into 3D voxels [13, 14, 15, 16, 17, 18], thereby applying sparse 3D convolutions to segment point cloud.

In this work, we summarize our contribution by combining the best of the first two approaches:

- We present a unified,hierarchical way to construct meaningful features from point cloud to improve semantic segmentation accuracy by designing bottom-up voxel feature aggregation from multiple scales and top-down pathway of pyramid-like structure.

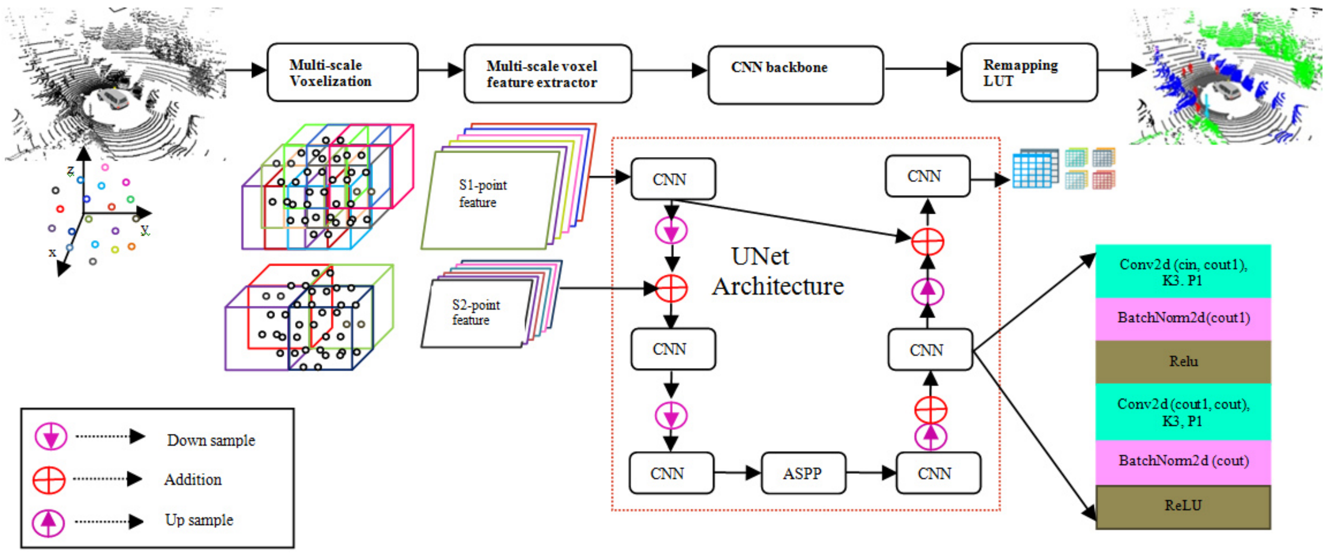

Figure 1. MSVCBASPP Architecture

- To improve semantic segmentation accuracy, class balanced weighted cross entropy loss is used in conjunction with Lovasz-Softmax loss [19].
- To boost segmentation performance, ASPP [7](Atrous spatial pyramid pooling) is used to concatenate feature maps generated by Atrous convolution with different dilation rates, so that the neurons in the output feature map contain multiple receptive field sizes to encode multi-scale information and simultaneously achieves a large receptive field size.
- To boost segmentation performance, DenseASPP [20] was used where each Atrous convolution layer only makes use of Atrous filters with reasonable dilation rate ($d \leq 24$), so that the neurons in the output feature map not only contains larger receptive fields but also dense feature maps.
- Improves training time of PolarNet [21] by incorporating learning-rate scheduler to the existing Adam optimizer [22]. This technique guides the network to accelerate training mechanism with fewer epochs without degrading accuracy as compared to baseline models.

All the above significant contributions are combined to form a model Multi-Scale Voxel Class Balanced ASPP(MSVCBASPP) network which has better semantic-segmentation performance over base-line model PolarNet [21]. The input to MSVCBASPP model is the rasterized image of the full LiDAR scan, and network outputs point-wise classification scores together with uncertainty measures.

Quantitative and qualitative experiments on the SemanticKITTI dataset [23] shows that the proposed MSVCBASPP model significantly outperforms other state-of-the-art networks in terms of point-wise segmentation accuracy while having much fewer parameters, lower computation time and real-time segmentation with only one GPU.

2. Related work

In this section, recent works in semantic segmentation of 3D point cloud data will be summarized. Recently great progress has been achieved in semantic segmentation of 3D LiDAR point clouds using deep neural networks [4, 5, 10, 11]. The core distinction between these advanced methods lies not only in the network design but also in the representation of the point cloud data. There are three common approaches of representing unstructured and unordered 3D LiDAR points.

2.1. Point-wise

Point-wise methods [24, 25] directly process the raw irregular 3D points without applying any additional transformation or pre-processing. Shared multi-layer perceptron based PointNet [24], the subsequent work PointNet++ [25], and superpoint graph SPG networks [26] are considered in this group. Although such methods are powerful on small point clouds, their processing capacity and memory requirement, unfortunately, becomes inefficient when it comes to the full 360 degree LiDAR scans. To accelerate point-wise operations, additional cues, e.g. from camera images, are employed as successfully introduced in [27].

2.2. Projection based

Projection based techniques convert 3D point cloud to 2D grids to enable the use of 2D CNNs. SqueezeSegv3 [11], SalsaNext [12], and RangeNet++ [10] utilize the spherical projection mechanism, which converts the point cloud to a frontal-view (range) image, and adopt the 2D convolution network on the pseudo image for segmentation. PolarNet [21] follows the bird-view projection, which projects point cloud data into small grids from the bird view and takes the height as a whole. Instead of partitioning points in a Cartesian coordinate system, they use a polar coordinate system for encoding point clouds.

2.3. 3D voxel partition

3D voxel partition is another routine of point cloud encoding [13, 14, 15, 16, 17, 18]. It converts a point cloud into 3D voxels. 3D U-Net [18] proposes voxel partition and 3D U-Net on biomedical data and shows successful application on difficult microscopic datasets. OccuSeg [14], SSCN [15] and SEGCloud [16] follow this line to utilize the voxel partition and apply 3D convolutions for LiDAR segmentation.

In the above three approaches, the closest work to ours is [21] which introduces a mix of both point-based approach and projection based approach. Here we introduce efficient techniques to improve PolarNet [21] model keeping accuracy and real-time applicability.

3. Method

Given a training dataset of N LiDAR scans. $\{(\mathrm{P}_i, L_i), i = 1, 2...N\}$,$\mathrm{P}_i \in R^{n_i * 4}$ is the point set containing n_i LiDAR points. Each row of P_i consists of four features representing one LiDAR point p namely $(x, y, z, reflection)$. (x, y, z) represents cartesian coordinate of the point relative to the scanner and $reflection$ represents the intensity of returning laser beam. $L_i \in Z^{n_i}$ contains the object labels for each point p_j in p_i. Our goal is to learn a segmentation model $f(., \theta)$ parameterized by θ so that the difference between the prediction $f(P_i)$ and L_i is minimized.

3.1. Pointcloud representation

As in [21], we represent point cloud to Polar bird-eye-view which has two fold advantages as compared to Cartesian coordinate. First, it evenly distributes points. Second, more balanced point distributions thereby reducing the burden on predictors.

Accordingly, we quantize points in Polar coordinate by converting each point on the XY plane to azimuth and radius with the sensor's location as the origin. We then assign points to grid cells based on their quantized azimuth and radius.

3.2. Pseudo image generation

Let l be a point in a point cloud with coordinates x, y, z and reflectance r. As a first step, the point cloud is discretized into an evenly spaced grid in the $\rho - \theta$ plane creating a set of uniform sized voxels. In the second step, points in each voxel are then augmented with $(\rho_c, \theta_c, z_c, \rho, \theta, z, x, y, r)$ where the subscript c denotes distance to the arithmetic mean of all points in the voxel. The augmented lidar point is now $D = 9$ dimensional. Next, we use a simplified version of PointNet [24] where, for each point, a linear layer is applied followed by BatchNorm [28] and ReLU [29]. Once encoded, the features are scattered back to the original voxel locations to create a pseudo-image of size (B, C, H, W) where H and W indicate the height and width of the pseudo-image.

3.3. CNN backbone

MSVCBASPP backbone has a UNet style of architecture consisting of top-down encoder network and bottom-up decoder network. Top-down encoder network produces features at increasingly small spatial resolution and a second bottom-up decoder network that performs upsampling and concatenation of the top-down features.

In this work, UNet architecture has been modified to adopt Feature Pyramid Network (FPN) [30] which is designed to combine multi-scale features of point cloud. Besides the bottom-up path used in multi-scale feature aggregation, we build a top-down path to efficiently construct a rich, hierarchical feature pyramid for multiple voxels features.

The top-down encoder backbone can be characterized by a series of five CNN blocks Block $(S = 2, L = 2, F)$. First CNN block is performed with a stride $S = 1$, then successive CNN blocks in top-down encoder is down sampled with a stride $S = 2$. Each block has $L = 2$, 3x3 2D ring convolution-layers with $F = 64, 128, 256, 512, 512$ output channels, each followed by BatchNorm [28] and a ReLU [29]. Total down-sampling factor of encoder backbone network is 16 w.r.t pseudo bird-eye-view image.

The bottom-up decoder network takes final features from each encoder block and are combined through up sampling and concatenation as follows. First, the features are up sampled, $UP(S_{in}, S_{out}, F)$ from an initial stride S_{in} to a final stride S_{out} using a bi-linear interpolation and then concatenated with encoder features. On the concatenated features, we apply CNN blocks. The final output features are a concatenation of all features that originated from different strides values.

3.4. Atrous spatial pyramid Pooling

It is a powerful tool inserted between encoder and decoder to enrich features of encoder by obtaining multi-scale context information. ASPP [7] segments objects at multiple scales by probing CNN features at multiple scales using Atrous convolutions in cascade/parallel to capture multi-scale context by adopting multiple Atrous rates. Also it captures global context using spatial-2D features to boost performance.

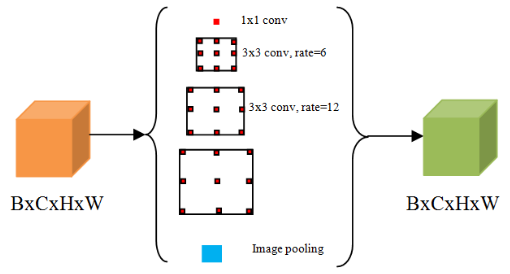

Figure 2. Atrous spatial pyramid pooling

After extracting feature maps from the backbone network, four parallel Atrous convolutions with different dilation rates are applied to handle segmenting the object at different scales. Image-level features are also applied to incorporate global context information by applying global average pooling on the last feature map of the backbone. After applying all the above operations parallelly, the results of each operation along the channel are concatenated and 1 x 1 convolution is applied to get larger receptive field feature maps.

3.5. Dense Atrous spatial pyramid Pooling

DenseASPP [20] combines the advantages of parallel and cascaded use of dilated convolutional layers and produces more scale features over a wider range. The name of DenseASPP is inherited from DenseNet [31] and also has the advantages of DenseNet like alleviating the gradient drop problem and has fewer parameters.

In DenseASPP, layers share information through skip connections and are interrelated. With dense connections, not only a denser feature pyramid is included, but also larger receptive fields are embedded to perceive wider environment information.

3.6. Class balanced loss function

Most of the publicly available datasets have an extreme imbalance between different classes. In the case of autonomous driving, traffic-sign/bicycle/motor-cycle appears less in the scene compared to road/car/truck/pedestrian. Such an imbalance between classes makes the network to be more biased towards classes that have more samples in training and thereby resulting in relatively poor segmentation results.

To boost the accuracy of the under-represented classes, we update the softmax cross-entropy loss with the smoothed

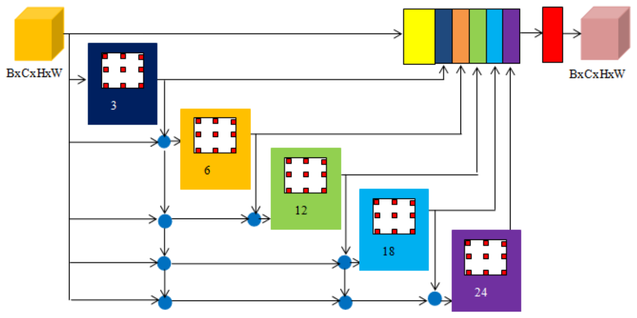

Figure 3. Dense Atrous spatial pyramid pooling

frequency of each class. Our class-balanced loss function is now weighted with the inverse square root of class frequency, defined as

$$L_{wce}(y, \hat{y}) = -\sum_{i} \alpha_i P(y_i) log(P(y_i)), \quad (1)$$

where

$$\alpha_i = 1/\sqrt{(}f_i).$$

where y and $\hat{y}$ are the true and predicted labels and f_i is the number of points belongs to i^{th} class. This technique strengthens the network each point that belongs to less frequent classes in the dataset. In addition to class balanced weighted cross entropy, we use Lovasz-Softmax loss [19] in the learning procedure to maximize the intersection-over-union (IoU) score, i.e. the Jaccard index which was followed in PolarNet [21] model.

4. Experimental results and analysis

We experiment our proposed MSVCBASPP on large-scale SemanticKITTI dataset [23] for autonomous driving. We conduct several experiments on various aspects: network performance on test/validation dataset, training/inference timing analysis, hyper parameter tuning based on grid size and state-of-art comparisons.

4.1. Dataset details

SemanticKITTI [23] is a point-level re-annotation of the LiDAR part of the famous KITTI dataset [32]. It has a total of 43551 scans sampled from 22 sequences collected in different cities in Germany. It has 104452 points per scan on average and each scan is collected by a single Velodyne HDL-64E laser scanner. There are 19 challenging classes in total. We follow SemanticKITTIs subset split protocol and use ten sequences for training, one for validation and the rest of them for testing. We present several baselines that have been presented with SemanticKITTI. We report the segmentation performance on the SemanticKITTI testing subset.

4.2. Evaluation metric

The performance of our model is measured on class level segmentation tasks by comparing each predicted point label with the corresponding ground truth annotation. As the primary evaluation metrics, we report intersection-over-union (IoU) results for each class as

$$IOU_i = \frac{|P_i \cap G_i|}{|P_i \cup G_i|}. \tag{2}$$

where P_i is the predicted point set of class i and G_i denotes the corresponding ground truth set, whereas $|.|$ returns the total number of points in a set. In addition, we report the average IoU score over all the nineteen classes.

4.3. Environment setup

To train MSVCBASPP, we employ the Adam optimizer [22] with the initial learning rate of 0.01, batch-size of 2 and ran the training for 50 epochs. We use TeslaK80 GPU 16GB for training and TitanV GPU 12GB to do inference. To increase the amount of training data, we add 25% probability each to randomly flip a point cloud along x, y and $x + y$ axes for data augmentation. Also we randomly rotate about the $z-$axis in the range of $[-5^o, +5^o]$. We use the same PolarNet[21] grid spaces to be $[distance : 0 \sim 70m, z : -3 \sim 1.5m]$. Also we set the respective grid sizes as $[480, 360, 32]$, $[360, 240, 32]$ and $[160, 120, 16]$.

4.4. Ablation study

Here we conduct multiple extensive ablation study of the proposed model on SemanticKITTI validation/test dataset considering accuracy and training/inference speed.

4.4.1 Analysis on SemanticKITTI validation dataset

In the Table-1, MSV, CB, PAM+CAM [33], ASPP1, ASPP2, DenseASPP1 and DenseASPP2 represents Multi-scale voxelization, Class balancing, Position attention module+channel attention module, Atrous spatial pyramid pooling with dilation rate $[6, 12, 18]$, Atrous spatial pyramid pooling with dilation rate $[8, 16, 24]$, Dense ASPP with dilation rate $[2, 4, 8, 12, 16]$ and Dense ASPP with dilation rate $[3, 6, 12, 18, 24]$.

BSL[21]	MSV	CB	PAM+CAM[33]	ASPP1	ASPP2	DenseASPP1	DenseASPP2	mIOU
Y	-	-	-	-	-	-	-	58.17
Y	Y	-	-	-	-	-	-	58.545
Y	Y	Y	-	-	-	-	-	58.714
Y	Y	Y	Y	-	-	-	-	58.162
Y	Y	Y	-	Y	-	-	-	60.4
Y	Y	Y	-	-	Y	-	-	**60.6**
Y	Y	Y	-	-	-	Y	-	58.942
Y	Y	Y	-	-	-	-	Y	59.213

Table 1. Model performance on SemanticKITTI validation dataset

From the Table-1, we observe that using ASPP2 in conjunction with MSV+CB, we can get maximum mIOU of 60.6 over performing baseline PolarNet [21] model by 1.5 mIOU. Also we observe that using DenseASPP2 in conjunction with MSV+ CB we can get maximum mIOU of 59.213 over performing baseline model by 1.1 mIOU.

4.4.2 Analysis on SemanticKITTI test dataset

From the Table-2, we observe that on test dataset the same MSV+CB+ASPP2 achieved maximum mIOU of 60.01 which was observed in the case of validation dataset with a 60.6 mIOU. Our best model MSV+CB+ASPP2 outperforms baseline PolarNet [21] model by a margin of 2.8 mIOU. Also, we observe that using DenseASPP2 in conjunction with MSV+CB we can get maximum mIOU of 58.95 over performing baseline model by 1.74 mIOU.

BSL[21]	MSV	CB	ASPP2	DenseASPP2	mIOU
Y	-	-	-	-	57.2
Y	Y	Y	-	-	58.6
Y	Y	Y	Y	-	**60.01**
Y	Y	Y	-	Y	58.95

Table 2. Model performance on SemanticKITTI test dataset

Ideally on both validation and test dataset, DenseASPP2 should perform better than ASPP2, but from the Table-[1,2] results are vice-versa. DenseASPP improves the semantic-segmentation performance, using DenseNet [31] architecture as a backbone whereas MSVCBASPP model uses UNet style of architecture. Hence MSVCBASPP authors believe DenseASPP2 didn't outperform ASPP2 based architecture.

4.4.3 Training time analysis on using LR schedulers

From the Table-3, we observe that using One-Cycle LR [34] in conjunction with Adam optimizer achieves faster training convergence only with 6 epochs to reach a maximum mIOU of 57.94 as compared to baseline [21] which uses only Adam optimizer without scheduler to reach 58.17 mIOU with 23 epochs.

BSL[21]	Cyclic[35]	CosAnn[36]	Expn	OneCycle[34]	mIOU	epochs
Y	-	-	-	-	58.17	23
Y	Y	-	-	-	54.18	16
Y	-	Y	-	-	57.91	18
Y	-	-	Y	-	51.18	10
Y	-	-	-	Y	57.94	**6**

Table 3. Model performance and training time analysis on using learning-rate schedulers

Using One-Cycle LR [34] scheduler, accuracy doesn't improve but this scheduler guides the network to train faster only with fewer epochs. This helped us to carry out a lot of experiments like ASPP1, ASPP2, PAM+CAM, DenseASPP1 and DenseASPP2.

4.4.4 Analysis of network performance and inference timings on varying grid sizes

In the Table-4, MODEL represents MSVCBASPP. Also G1,G2,G3,G4 represents tunable grid sizes (480,360,32), (360,240,32), (160,120,16) and (80,60,8).

BSL[21]	Model-G1	Model-G2	Model-G3	Model-G4	mIOU	Runtime(ms)
Y	-	-	-	-	58.17	48
-	Y	-	-	-	**60.6**	49
-	-	Y	-	-	57.97	42
-	-	-	Y	-	54.12	**29**
-	-	-	-	Y	43.92	24

Table 4. Model performance and inference timing analysis for various grid sizes on validation dataset

From the Table-4, we observe that MODEL-G1 achieves maximum mIOU of 60.6 with run-time around 49ms per frame. Whereas our lighter-weight MODEL-G3 achieves 54.12 mIOU consuming only 29ms which strikes balance between accuracy, speed and suitable for real-time vehicle testing. Also we observe that as grid size reduces further, performance of MODEL-G4 severely degrades.

4.4.5 Analysis of network params and FLOPS count

From the Table-5 below, we observe that our MSVCBASPP model parameter and FLOPS count is increased by the addition of multi-scale voxelization along with ASPP modules as compared to PolarNet [21] model but the run-time difference between the models are very marginal.

Model	mIOU	Runtime(ms)	Params(M)	FLOPS(GMAC)
BSL[21]	58.17	**48**	**13.6**	**135**
MSVCBASPP(ours)	**60.6**	49	22.4	142

Table 5. Model parameter and FLOPS count analysis

4.5. State-of-art comparison analysis

Figure-4 shows visualization results of MSVCBASPP model on random sequence of SemanticKITTI test dataset. Table-6 shows the performance comparison between our approach and multiple baselines on SemanticKITTI test dataset. The results demonstrate that our model outperforms many state-of-art methods and yet remain competitive in SemanticKITTI leaderboard at the time of submission.

5. Conclusion

In this paper, we propose efficient techniques to significantly improve model performance not only in speed and accuracy but also guides the model to train faster with fewer epochs. Our empirical experiments on SemanticKITTI dataset based on MSVCBASPP model shows that the efficient techniques applied consistently improves LiDAR semantic segmentation model which is comparable to state-of-art techniques in SemanticKITTI leaderboard. Moreover our experiments also shows that our network can balance between speed/accuracy with high-resolution-grid model(60.6 mIOU, 21fps) and low-resolution-grid model(54.01 mIOU, 35fps) using only one GPU.

References

[1] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016.

[2] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals using stereo imagery for accurate object class detection. *IEEE transactions on pattern analysis and machine intelligence*, 40(5):1259–1272, 2017.

[3] R Amadeo. Google's waymo invests in lidar technology, cuts costs by 90 percent. *Ars Technica, Jan*, 10, 2017.

[4] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6411–6420, 2019.

[5] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020.

[6] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[7] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[8] Yuhui Yuan, Xilin Chen, and Jingdong Wang. Object-contextual representations for semantic segmentation. *arXiv preprint arXiv:1909.11065*, 2019.

[9] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv:1904.04514*, 2019.

[10] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. Rangenet++: Fast and accurate lidar semantic segmentation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4213–4220. IEEE, 2019.

[11] Chenfeng Xu, Bichen Wu, Zining Wang, Wei Zhan, Peter Vajda, Kurt Keutzer, and Masayoshi Tomizuka. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. *arXiv preprint arXiv:2004.01803*, 2020.

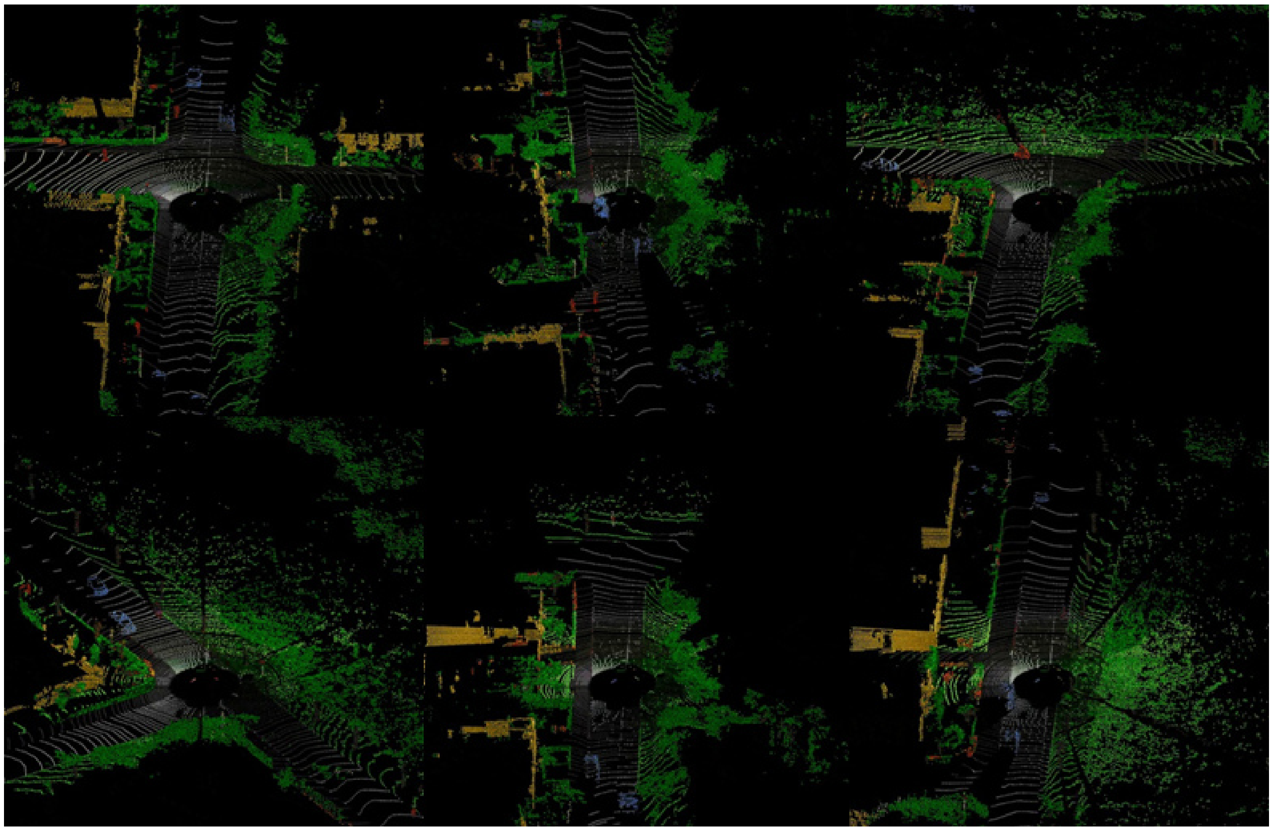

Figure 4. Results visualization on SemanticKITTI test dataset

Method	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign	mIOU
PointNet[24]	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	14.6
PointNet++[25]	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
SPLATNET[37]	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
TangentConv[38]	86.8	1.3	12.7	11.6	10.2	17.1	20.2	0.5	82.9	15.2	61.7	9.0	82.8	44.2	75.5	42.5	55.5	30.2	22.2	35.9
RandLA-Net[5]	94.2	26.0	25.8	40.1	38.9	49.2	48.2	7.2	90.7	60.3	73.7	38.9	86.9	56.3	81.4	61.3	66.8	49.2	47.7	53.9
RangeNet53++[10]	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
3D-MiniNet[39]	90.5	42.3	42.1	28.5	29.4	47.8	44.1	14.5	91.6	64.2	74.5	25.4	89.4	60.8	82.8	60.8	66.7	48.0	56.6	55.8
SqueezeSeg-V3[11][15]	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
SalsNext[12]	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
Cylinder3D[40]	96.1	54.2	47.6	38.6	45.0	65.1	63.5	13.6	91.2	62.2	75.2	18.7	89.6	61.6	85.4	69.7	69.3	62.6	64.7	61.8
SPV-NAS[41]	97.2	50.6	50.4	56.6	58.0	67.4	67.1	50.3	90.2	67.6	75.4	21.8	91.6	66.9	86.1	73.4	71.0	64.3	67.4	67.0
PolarNet[21]	93.5	44.3	38.3	25.6	30.7	52.0	44.5	40.6	89.7	61.8	72.1	7.5	91.6	63.4	84.2	63.7	69.0	55.4	60.7	57.2
MSVCB(ours)	93.4	51.3	41.9	43.0	32.5	57.1	49.4	16.2	89.8	59.1	73.4	23.7	90.4	93.4	81.6	62.6	66.2	52.3	59.2	58.2
MSVCBASPP(ours)	94.1	52.5	45.3	24.5	37.4	58.4	53.7	46.0	91.3	58.7	73.4	24.0	89.6	60.0	81.9	24.5	64.2	55.4	60.4	60.01

Table 6. State-of-art model performance comparison on SemanticKITTI test dataset

[12] Tiago Cortinhal, George Tzelepis, and Eren Erdal Aksoy. Salsanext: Fast semantic segmentation of lidar point clouds for autonomous driving. *arXiv preprint arXiv:2003.03653*, 2020.

[13] Tai Wang, Xinge Zhu, and Dahua Lin. Reconfigurable voxels: A new representation for lidar-based point clouds. *arXiv preprint arXiv:2004.02724*, 2020.

[14] Lei Han, Tian Zheng, Lan Xu, and Lu Fang. Occuseg: Occupancy-aware 3d instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2940–2949, 2020.

[15] Xinge Zhu, Yuexin Ma, Tai Wang, Yan Xu, Jianping Shi, and Dahua Lin. Ssn: Shape signature networks for multiclass object detection from point clouds. *arXiv preprint arXiv:2004.02774*, 2020.

[16] Lyne Tchapmi, Christopher Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. Segcloud: Semantic segmentation of 3d point clouds. In *2017 international conference on 3D vision (3DV)*, pages 537–547. IEEE, 2017.

[17] Benjamin Graham, Martin Engelcke, and Laurens Van

Der Maaten. 3d semantic segmentation with submanifold sparse convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9224–9232, 2018.

[18] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention*, pages 424–432. Springer, 2016.

[19] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4413–4421, 2018.

[20] Maoke Yang, Kun Yu, Chi Zhang, Zhiwei Li, and Kuiyuan Yang. Denseaspp for semantic segmentation in street scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3684–3692, 2018.

[21] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, and Hassan Foroosh. 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*, pages 9601–9610, 2020.

[22] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[23] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Jurgen Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9297–9307, 2019.

[24] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[25] 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.

[26] Loic Landrieu and Martin Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4558–4567, 2018.

[27] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 918–927, 2018.

[28] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[29] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[30] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[31] Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. *arXiv preprint arXiv:1404.1869*, 2014.

[32] 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.

[33] Jun Fu, Jing Liu, Haijie Tian, Yong Li, Yongjun Bao, Zhiwei Fang, and Hanqing Lu. Dual attention network for scene segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3146–3154, 2019.

[34] Leslie N Smith and Nicholay Topin. Super-convergence: Very fast training of neural networks using large learning rates. In *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, volume 11006, page 1100612. International Society for Optics and Photonics, 2019.

[35] Leslie N Smith. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472. IEEE, 2017.

[36] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[37] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2530–2539, 2018.

[38] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3887–3896, 2018.

[39] Iñigo Alonso, Luis Riazuelo, Luis Montesano, and Ana C Murillo. 3d-mininet: Learning a 2d representation from point clouds for fast and efficient 3d lidar semantic segmentation. *arXiv preprint arXiv:2002.10893*, 2020.

[40] Hui Zhou, Xinge Zhu, Xiao Song, Yuexin Ma, Zhe Wang, Hongsheng Li, and Dahua Lin. Cylinder3d: An effective 3d framework for driving-scene lidar semantic segmentation. *arXiv preprint arXiv:2008.01550*, 2020.

[41] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. *arXiv preprint arXiv:2007.16100*, 2020.