Multimodal 3D Semantic Segmentation

Fabian Duerr

Vision and Fusion Laboratory
Institute for Anthropomatics
Karlsruhe Institute of Technology (KIT), Germany
fabian.duerr@audi.de

Technical Report IES-2019-06

Abstract

Understanding and interpreting a scene is a key task of environment perception for autonomous driving, which is why autonomous vehicles are equipped with a wide range of different sensors. Semantic Segmentation of sensor data provides valuable information for this task and is often seen as key enabler. In this report, we're presenting a deep learning approach for 3D semantic segmentation of lidar point clouds. The proposed architecture uses the lidar's native range view and additionally exploits camera features to increase accuracy and robustness. Lidar and camera feature maps of different scales are fused iteratively inside the network architecture. We evaluate our deep fusion approach on a large benchmark dataset and demonstrate its benefits compared to other state-of-the-art approaches, which rely only on lidar.

1 Introduction

One of the key challenges of autonomous driving is the understanding of the vehicle's environment. Therefore, autonomous vehicles are equipped with a wide range of sensor modalities, usually including, camera, lidar, radar and ultrasonic

sensors. With different complementary sensors available, shortcomings of an individual sensor type can be compensated by other sensor types, increasing accuracy and robustness. In this work, we focus on camera and lidar sensors.
Understanding and interpreting a scene is a key task of environment perception for autonomous driving, which makes semantic segmentation of sensor data valuable. For camera images, assigning a class label to every image pixel has been addressed very successfully with Convolutional Neural Networks (CNNs) over the past years, achieving impressive results on road and urban scenes [5]. When dealing with 3D lidar point clouds however, the first challenge is a proper representation, enabling the application of CNNs. One possibility is the lidar's native range view, which has shown promising results [15, 16]. This allows the application of established image segmentation architectures.
Having different sensors available with an overlapping field of view, allows for approaches that fuse the data of different sensors to improve the robustness and overall accuracy. When addressing the fusion of camera and lidar data, some challenges arise. One is a substantial difference in their resolution and another is their considerable difference in measurements and sensor space. While a camera observes brightness values resulting in an image, a lidar measures the distance to its environment, generating a sparse 3D point cloud. Additionally, different fusion strategies must be considered. Following [4], these are the fusion of the sensor data (early fusion), the fusion of the predictions for lidar and camera data (late fusion) or the fusion of the featues maps inside a CNN (deep fusion). In this work, we propose a deep fusion approach, applied to the range view representation, which makes use of camera and lidar data to calculate a semantic segmentation of lidar point clouds. The contributions of this work are twofold:

- First, we propose a fusion module, which takes camera and lidar features, transforms them into a common space and fuses them afterwards.
- Second, we propose a fusion architecture building upon the fusion modules and apply them iteratively throughout our network, following the idea of iterative deep aggregation [26]. This way, we are able to fuse aggregated features of both sensors at different scales and maximize the fused information

2 Related work

2.1 2D Semantic segmentation

The success of deep learning applied for scene parsing and semantic segmentation [13, 21, 8] is closely related to its success in classical image classification [22, 10, 7]. One widely used approach are Fully Convolutional Neural Networks (FCNN) [13], which calculate a pixel-wise prediction for a given image in an end-to-end fashion. [13] replaced the fully connected layers of common classification architectures with 1x1-convolutions, thereby replacing the original image classification with a pixel-wise classification.
One main challenge, recent works have focused on, is the loss of spatial resolution while aggregating information. It is of great importance to capture the global context of a scene as well as fine local structures. DeepLabv3 [3, 2] addresses this by 'atrous' convolutions, which increase the size of the receptive fields without reducing resolution or increasing filter sizes. 'Atrous' convolutions with different rates are employed in parallel to exploit context at different scales. In [26], an aggregation architecture is presented, which the authors call deep layer aggregation (DLA), also targeting the challenge of extracting meaningful semantic features while preserving spatial information. PSPNet [29] combines local and global context by a pyramid pooling module, which aggregates the global context at different scales and appends it to the original feature maps. OcNet [27] adapts the idea of the pyramid pooling module and multiscale 'atrous' convolutions by introducing an object context module, which exploits object context at different scales, instead of spatial context.

2.2 3D Semantic segmentation

When addressing semantic segmentation of 3D point clouds with CNNs, the first thing to consider is the representation of the point clouds. In recent works, multiple different representations are proposed. PointNet [18] uses the raw and unstructured point clouds directly as input by applying pointwise 1x1-convolutions and a symmetric operation for feature aggregation. Because a single global feature aggregation limits the ability to capture spatial relations, the

authors proposed PointNet++ [19], which applies individual PointNets to local regions and aggregates the resulting local features in a hierarchical fashion. [23] converts the point clouds into a voxel grid and applies a 3D-FCNN, followed by a Conditional Random Field (CRF) to refine the results. A bird's eye view (BEV) with the vertical axis as feature channel is used by [28] to retrieve a 2D representation of the point clouds. Having a 2D representation, they're using the U-Net architecture [21], known from image segmentation.
When working with point clouds generated by a lidar sensor, the range view is another possibility of representation. SqueezeSeg [24] was one of the first works using the range view for a segmentation task. Their goal was the segmentation of road objects, with an improved version released in [25]. Another approach is RangeNet++ [16], which employs the DarkNet53 backbone [20] for full semantic segmentation. [14] proposed LaserNet, which uses the range view as input for object detection, while one of their intermediate results is a semantic segmentation of the input. Their architecture is based on deep layer aggregation. Transforming the point cloud into its range view and applying established 2D image segmentation architectures mostly outperforms other forms of representations while being faster. Therefore, our work also builds upon the range view representation.

2.3 Multimodal 3D semantic segmentation

Multi-sensor fusion architectures using camera and lidar mostly focus on object detection [4, 17, 11, 12, 15]. Only [15] also tackles the task of 3D semantic segmentation, using the range view as input representation. Camera image feature maps, extracted by three ResNet blocks [7], and extracted lidar feature maps from the range view are concatenated and passed to a LaserNet, which serves as DLA for the semantic segmentation. In contrast to applying early fusion and fusing the RGB values with the range view, this approach aggregates camera image information first, using the original usually much higher resolution of the camera image. This deep fusion allows for more information being preserved and exploited for the semantic segmentation of the lidar point cloud. While considerably improving the mean Intersection over Union over all classes (mIoU) on distant content $(+5.19)$, the overall improvements are rather small $(+0.25)$.

We're also using deep layer aggregation and the full camera image resolution for deep fusion of camera and lidar. In contrast to [15], which fuses the features before applying their DLA network (LaserNet), we're applying a DLA network to both, the lidar range view and the camera image, separately but fuse both networks following iterative deep aggregation [26]. As a result, our deep fusion approach is able to aggregate and use more information from the camera for the semantic segmentation of the lidar point cloud.

3 Iterative deep fusion and aggregation

In this section, we present our range view input representation, our fusion module and the network architecture, used for the fusion of the lidar and camera input.

3.1 Range view

Commonly used lidar sensors usually observe their environment by spinning a set of vertically stacked lasers around their vertical axis. The position of a laser in this stack is often referred to as channel, corresponding to an elevation angle. The Velodyne HDL-64E, used to record the SemanticKitti dataset [1, 6], has 64 channels, an azimuth resolution of approximately $0.17°$ and an elevation resolution of $^1/_3{}^\circ$ for the upper and $^1/_2{}^\circ$ for the lower half of the lasers. The sensor provides measurements $\boldsymbol{o}_i = (c_i, \phi_i, r_i, e_i)$, with channel id c_i, azimuth angle ϕ_i, measured distance r_i and reflectance e_i. The corresponding 3D points are

$$\boldsymbol{p}_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = \begin{pmatrix} r_i \cos(\theta_i) \cos(\phi_i) \\ r_i \cos(\theta_i) \sin(\phi_i) \\ r_i \sin(\theta_i) \end{pmatrix}, \tag{3.1}$$

omitting correction factors. The elevation angle θ_i is derived from the sensor configuration and the channel id c_i.
We generate a range view by mapping every point or measurement to a row and column index. Having measurements from a Velodyne HDL-64E, the row and column indices are calculated by using the channel as row index and discretizing

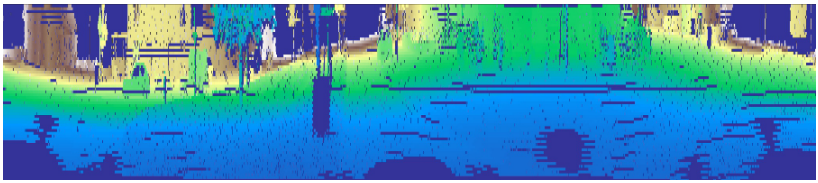

Figure 3.1: range view showing the lidar depth measurements.

the azimuth angle. If only the 3D points $\boldsymbol{p}_i$ are provided, the azimuth and elevation angle are given by

$$\phi_i = -\text{arctan2}(y_i, x_i) \quad \text{and} \quad \theta_i = \arcsin\left(\frac{z_i}{r_i}\right). \tag{3.2}$$

Finally, for a range view resolution of $h \times w$, the image coordinates $\boldsymbol{u}_i^{\text{li}} = (u_i^{\text{li}}, v_i^{\text{li}})$ are

$$u_i^{\text{li}} = \begin{cases} \left\lfloor 0.5 \cdot h \cdot \frac{\theta_i - \theta_{\text{up}}}{\theta_{\text{mid}} - \theta_{\text{up}}} \right\rfloor & \theta_i \geq \theta_{\text{mid}} \\ \left\lfloor 0.5 \cdot h \cdot \left(1 + \frac{\theta_i - \theta_{\text{mid}}}{\theta_{\text{down}} - \theta_{\text{mid}}}\right) \right\rfloor & \theta_i < \theta_{\text{mid}} \end{cases}, \tag{3.3}$$

$$v_i^{\text{li}} = \left\lfloor 0.5 \cdot \left(1 + \frac{\phi_i}{\pi}\right) \cdot w \right\rfloor, \tag{3.4}$$

with a vertical field of view $\theta_{\text{fov}} = \theta_{\text{up}} - \theta_{\text{down}} = 2° - (-24.8°) = 26.8°$ and the border angle between the two vertical resolutions $\theta_{\text{mid}} = -{}^{26}/_{3}{}^{\circ}$. Following this, we're mapping the input measurements r, e, x, y and z to the 2D range view, receiving a $5 \times h \times w$ input tensor $\mathbf{R}$. The depth channel (r) is visualized in Fig. 3.1.

Ego motion, uncertainty and non-uniformity of the angles can lead to mapping collisions. As a result, more than one point is mapped to the same range view pixel. This implies not only a loss of information but also missing predictions for the shadowed points. The latter isn't an issue for object detection, for semantic segmentation however, it has to be considered. Therefore, a post-processing step based on the labeled points is required to compute class labels for the shadowed points. Following the simplest one, we assign the same label to all

measurements projected on the same range view pixel. Another approach is based on k-nearest neighbor [16]. We will investigate the post-processing step in future work. In this work, we're focusing on the feature fusion.

3.2 Feature transformation and fusion

A crucial part of our work is the feature fusion, which fuses the lidar and camera features. We're choosing the range view as our reference system and project camera features into it. The inverse projection, from lidar to camera, is mathematically given by the equation

$$\begin{pmatrix} u_i^{\text{cam}} \\ v_i^{\text{cam}} \\ 1 \end{pmatrix} = \mathbf{K} \cdot \mathbf{T}_{\text{li2cam}} \cdot \begin{pmatrix} \boldsymbol{p}_i \\ 1 \end{pmatrix}, \tag{3.5}$$

with the camera matrix $\mathbf{K}$ and transformation matrix from lidar to camera $\mathbf{T}_{\text{li2cam}}$. The calculated pixel indices define the correspondence between 3D points and camera pixels. For this correspondence being still valid after scaling the range view by β or the camera image by α, the following extensions are made

$${}^{\alpha}\boldsymbol{u}_i^{\text{cam}} = \begin{pmatrix} \lfloor u_i^{\text{cam}} \cdot \alpha \rfloor \\ \lfloor v_i^{\text{cam}} \cdot \alpha \rfloor \end{pmatrix} \quad \& \quad {}^{\beta}\boldsymbol{u}_i^{\text{li}} = \begin{pmatrix} \lfloor u_i^{\text{li}} \cdot \beta \rfloor \\ \lfloor v_i^{\text{li}} \cdot \beta \rfloor \end{pmatrix}, \quad \text{with } \alpha, \beta \in [0, 1]. \tag{3.6}$$

Given scalable projection indices, we're now able to project camera features $\mathbf{I}^{\alpha}$ into the range view $\mathbf{R}^{\beta}$, following

$$\mathbf{R}^{\beta}[\,{}^{\beta}\boldsymbol{u}_i^{\text{li}}] \coloneqq \mathbf{I}^{\alpha}[\,{}^{\alpha}\boldsymbol{u}_i^{\text{cam}}]. \tag{3.7}$$

This is a fixed, geometrically motivated mapping, considering only one location per 3D point in the camera feature maps. To capture more context and to compensate errors in the calibration, we apply a learnable function $F_{\boldsymbol{w}}$ before performing the fixed projection, resulting in

$$\mathbf{I}_F^{\alpha} = F_{\boldsymbol{w}}(\mathbf{I}^{\alpha}) \quad \text{and} \quad \mathbf{R}_{\boldsymbol{w}}^{\beta}[\,{}^{\beta}\boldsymbol{u}_i^{\text{li}}] \coloneqq \mathbf{I}_F^{\alpha}[\,{}^{\alpha}\boldsymbol{u}_i^{\text{cam}}]. \tag{3.8}$$

The fusion module shown in Fig. 3.2 builds upon this to implement the camera feature transformation. We're using a 3x3 convolution followed by Batch Norm [9] and ReLu as learnable function $F_{\boldsymbol{w}}$. The projected camera features and the lidar features are concatenated and fused by ResNet blocks.

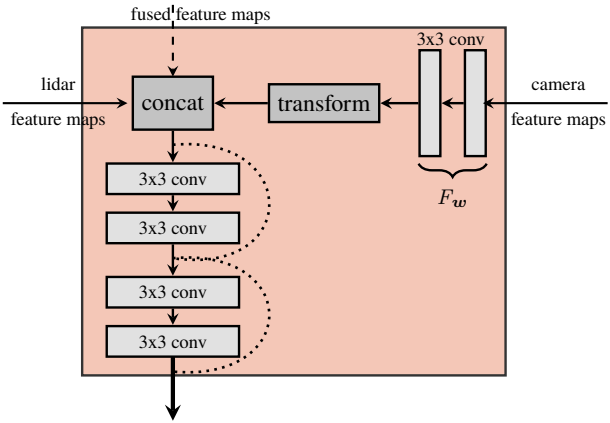

Figure 3.2: The main building block of our architecture. The fusion module transforms the camera features into the lidar range view. Afterwards, lidar feature maps, camera feature maps and optionally fused features maps from the stage before are fused.

3.3 Network architecture

Our proposed network architecture is shown in Fig. 2.3 and has three main components. First, a DLA network called Lidar-Net (I) for processing the lidar range view and calculating lidar features. It follows the proposed architecture of [14], which itself is based on [26]. By using a DLA architecture, we ensure to efficiently aggregate multi-scale lidar features. The second component is another DLA network (II) with the same architecture for processing the camera image. Additionally, we downsample the camera image before applying the DLA network. The resolution of the camera image is much higher than of the lidar image, so the induced loss in spatial information is small, whereas the aggregated semantic information are considerably improved. We follow the ResNet architecture and downsample the camera image with a strided convolution and max pooling by a factor of four. This also decreases the run time and memory requirements. The last component are fusion blocks (III), which apply the previously presented feature transformation and fusion. They

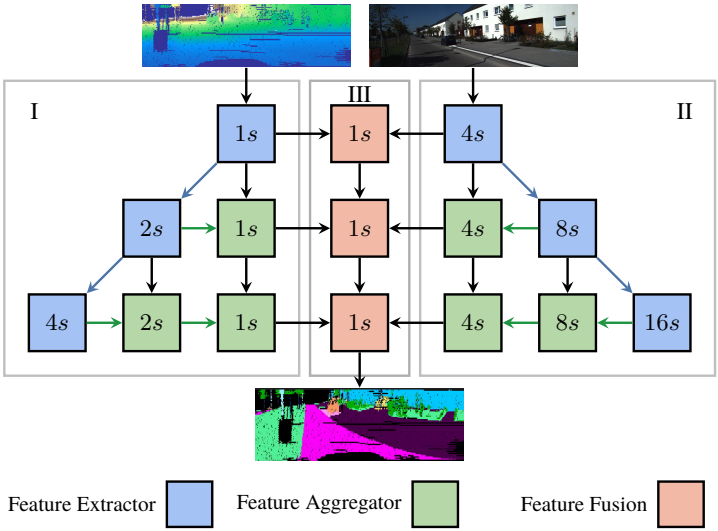

Figure 3.3: Our proposed fusion architecture, which fuses the lidar and camera features iteratively, following the idea of iterative deep aggregation [26]. The labels indicate the output stride of the individual blocks. We use the same network parameters for (I) and (II) as [14].

follow the idea of a feature aggregator except that they transform and aggregate features of different sensors instead of different scales of one sensor.

4 Experiments

4.1 SemanticKitti

We're evaluating our approach on the SemanticKitti dataset [1, 6], which contains labels for 19 classes for the single scan benchmark. A total of 22 labeled sequences results in 43552 labeled scans. The official split allocates sequences 0-10 for training and sequences 11-21 for testing, for which the labels haven't been published. However, the official benchmark doesn't support the usage of the camera images, meaning for our evaluation, only the sequences

with published labels 0-10 can be used. Therefore, were excluding sequences 02, 06 and 10 from training and validation and use them only in the end for testing. This results in 6963 frames for testing and 16238 for training and validation. We follow the official evaluation metric and report the mean Intersection-over-Union (mIoU). For our approach, only the lidar scan parts overlapping with the camera's field of view in the front of the car can be used.

4.2 Implementation details

Our training starts with an initial learning rate of 10^{-4}, which is then multiplied in each training iteration it by $10^{\frac{-2 \cdot \text{it}}{\text{it}_{\text{max}}}}$. Thereby, the learning rate exponentially decreases by $1/100$ during training. We train our network for $50k$ iteration with a batch size of 40. To improve generalizability and reduce overfitting, we're using random crops of the whole $360°$ lidar scan for training the lidar net. Although the crop is random, it follows the constraint, that the overlapping field of view with the camera has to be fully inside the crop of size 64×1536. The fusion modules finally crop the resulting lidar feature maps exactly to the overlapping field of view. Additionally, we apply random flipping horizontally to the lidar and camera images.
To counteract the class imbalance, we're using a class-balanced cross entropy loss for the final output as well as the auxiliary loss. The latter is used on the final feature map of the Lidar-Net. Following the proposed settings of PSPNet [29], we're weighting the auxiliary loss by 0.4

4.3 Results

We evaluate our approach and present the improvements gained by the fusion of lidar and camera image features. Therefore, we compare the results of our deep fusion architecture, called Fusion-Net, to Lidar-Net, which uses only the lidar scans. The results of both approaches are shown in Tab. 4.1. Overall, our fusion approach outperforms Lidar-Net by a considerable margin, and also the majority of the individual classes considerably benefit from the deep fusion approach.

Approach	road	sidewalk	parking	other-ground	building	car	truck	bicycle	motorcycle	other-vehicle
Lidar-Net	93.1	76.8	**56.1**	**3.4**	67.1	81.7	**42.0**	23.2	39.8	29.0
Fusion-Net	**93.2**	**77.0**	55.9	0.4	**74.0**	**82.0**	37.8	**26.4**	**43.1**	**29.1**

Approach	vegetation	trunk	terrain	person	bicyclist	motorcyclist	fence	pole	traffic sign	mIoU
Lidar-Net	78.0	58.1	67.2	35.6	**11.8**	**2.4**	57.2	36.4	39.9	47.3
Fusion-Net	**81.4**	**65.8**	**72.0**	**42.7**	11.0	0.3	**59.4**	**49.6**	**45.6**	**49.8**

Table 4.1: Comparison of the results of our deep fusion architecture and the purely lidar based Lidar-Net

5 Conclusion and Outlook

In this work, we've presented a deep learning approach for semantic segmentation of 3D lidar point clouds. Our approach uses a range view representation of the lidar scans, enabling the application of established image segmentation approaches. Furthermore, we use camera image feature maps of different scales and iteratively fuse them inside our network with the lidar feature maps. Our experiments underline the advantages of our deep fusion approach, which outperforms a lidar-only approach by a considerable margin in terms of the mIoU. Also, most of the individual classes considerably benefit from the fusion. For the future, we plan to further improve our fusion modules and thereby increase the benefits of our fusion architecture. We're also planning a more in depth analysis of the benefits of fusing camera and lidar data for 3D semantic segmentation.

References

[1] Jens Behley et al. “SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences.” In: *IEEE International Conference on Computer Vision (ICCV)*. 2019.

[2] Liang-Chieh Chen et al. “DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40 (2016), pp. 834–848.

[3] Liang-Chieh Chen et al. “Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs”. In: *International Conference on Learning Representations (ICLR)* (2015).

[4] Xiaozhi Chen et al. “Multi-view 3D Object Detection Network for Autonomous Driving”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 6526–6534.

[5] Marius Cordts et al. “The Cityscapes Dataset for Semantic Urban Scene Understanding”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 3213–3223.

[6] Andreas Geiger, Philip Lenz, and Raquel Urtasun. “Are we ready for autonomous driving? The KITTI vision benchmark suite”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2012), pp. 3354–3361.

[7] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 770–778.

[8] Kaiming He et al. “Mask R-CNN”. In: *The IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.

[9] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *ArXiv* abs/1502.03167 (2015).

[10] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[11] Jason Ku et al. “Joint 3D Proposal Generation and Object Detection from View Aggregation”. In: *IEEE International Conference on Intelligent Robots and Systems (IROS)* (2017), pp. 1–8.

[12] Bo Li, Tianlei Zhang, and Tian Xia. “Vehicle Detection from 3D Lidar Using Fully Convolutional Network”. In: *ArXiv* (2016).

[13] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 3431–3440.

[14] Gregory P. Meyer et al. “LaserNet: An Efficient Probabilistic 3D Object Detector for Autonomous Driving”. In: *ArXiv* abs/1903.08701 (2019).

[15] Gregory P. Meyer et al. “Sensor Fusion for Joint 3D Object Detection and Semantic Segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2019.

[16] A. Milioto et al. “RangeNet++: Fast and Accurate LiDAR Semantic Segmentation”. In: *IEEE International Conference on Intelligent Robots and Systems (IROS)*. 2019.

[17] Charles Ruizhongtai Qi et al. “Frustum PointNets for 3D Object Detection from RGB-D Data”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018), pp. 918–927.

[18] Charles Ruizhongtai Qi et al. “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 77–85.

[19] Charles Ruizhongtai Qi et al. “PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space”. In: *NIPS*. 2017.

[20] Joseph Redmon and Ali Farhadi. “YOLOv3: An Incremental Improvement”. In: *ArXiv* abs/1804.02767 (2018).

[21] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)* Vol.9351 (2015), pp. 234–241.

[22] Olga Russakovsky et al. “Imagenet large scale visual recognition challenge”. In: *International journal of computer vision* 115.3 (2015), pp. 211–252.

[23] Lyne P. Tchapmi et al. “SEGCloud: Semantic Segmentation of 3D Point Clouds”. In: *International Conference on 3D Vision (3DV)* (2017), pp. 537–547.

[24] Bichen Wu et al. “SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud”. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2017), pp. 1887–1893.

[25] Bichen Wu et al. “SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud”. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2018), pp. 4376–4382.

[26] Fisher Yu et al. “Deep Layer Aggregation”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 2403–2412.

[27] Yuhui Yuan and Jingdong Wang. “OCNet: Object Context Network for Scene Parsing”. In: *ArXiv* abs/1809.00916 (2018).

[28] Chris Zhang, Wenjie Luo, and Raquel Urtasun. “Efficient Convolutions for Real-Time Semantic Segmentation of 3D Point Clouds”. In: *International Conference on 3D Vision (3DV)* (2018), pp. 399–408.

[29] Hengshuang Zhao et al. “Pyramid Scene Parsing Network”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 6230–6239.