Optimizing Panoptic Segmentation on LiDAR Point Cloud data

Soham Aserkar, Anuj Pai Raikar, Swapneel Wagholikar, Deepak Nagle
Robotics Engineering Department
Worcester Polytechnic Institute Worcester, MA, USA

***Abstract*—Panoptic segmentation of point clouds is a crucial task for autonomous vehicles to comprehend their surroundings when using their highly accurate and reliable LiDAR sensors. Recent developments in panoptic segmentation have rekindled the interest of the scientific community in integrating semantic (things) and instance (stuff) segmentation effectively. RangeNet is one of the most advanced encoder-decoder-based algorithms for semantic segmentation of LiDAR-based 3-D point cloud data. In this work, we aimed at implementing a Mask-RCNN-based instance head to a RangeNet backbone to run it in parallel with the semantic head. Additionally, the semantic and instance heads are unified by pixel-wise classification to create a parameter-free panoptic head. It was expected to give results with faster inference, however, it is observed that it resulted in a complex architecture due to processing constraints, due to which we were able to achieve results till Semantic Segmentation.**

***Index Terms*—Panoptic Segmentation, Semantic Segmentation, Instance Segmentation, RangeNet, Mask R-CNN**

I. Introduction

There have been continuous developments in the panoptic segmentation tasks, which involve both semantic and instance segmentation. The semantic segmentation (stuff) assigns semantic labels to each stuff class. The instance segmentation identifies and classifies each occurrence within the things class. It is used for detecting and demarcating various object boundaries. This is crucial in areas like autonomous driving. In this case, the LIDAR-based data has to be processed in real-time to classify road class from trees, walls, and so on. These interesting applications made us select this topic for our project. The data which we worked on is the SemanticKITTI[1] which is a readily available LIDAR-based dataset.

The developments in Panoptic Segmentation were made possible by simple yet powerful baseline methods, such as Fully Convolutional Networks (FCN) [2], fast R-CNN [3], faster R-CNN [4], and Mask R-CNN [5]. These algorithms are robust and show higher flexibility, as well as provide a basis for much of the contemporary developments in these areas.

In this project, our aim is to implement semantic and instance head as an extension to the RangeNet backbone. This will be followed by a common panoptic head which renders the results of both instance and semantic segmentation heads. Our instance-head uses the Mask R-CNN architecture which involves a 2 stage framework: the first stage scans the image and generates proposals (areas likely to contain an object), and the second stage classifies the proposals and generates bounding boxes and masks. For each pixel in the panoptic head, if it belongs to the stuff, the goal of the panoptic segmentation system is simply to predict the class label within the stuff class. Otherwise, the system has to decide which instance and which class of things it belongs to. The challenge of this task lies in the fact that the system must give a definite answer for each pixel.

II. Research Contributions

We aim to implement the following modifications to the existing Panoptic Segmentation architecture:

1. Extension of modified RangeNet baseline to implement a semantic head and Mask R-CNN-based instance head.

2. Panoptic head implementation via pixel-wise classification to get the final panoptic output.

III. Related Work

Semantic Segmentation: Semantic segmentation tasks for self-driving cars using images have made incredible progress in the past decade due to the emergence of deep neural networks and the availability of increasingly large-scale datasets for the task, such as Cityscapes [15] and SemanticKITTI[16], SqueezeSeg and SqueezeSegV2 [17], [18], by Wu et al., utilize a spherical projection method of the point cloud to enable the usage of 2D convolutions. Also, a lightweight FCN is applied for the task along with a conditional random field (CRF) to smooth the results. PolarNet [19] projects the 3D point cloud data in the birds-eye view format and uses ring convolutions on the radially defined grids. The current state-of-the-art for semantic segmentation is the RangeNet, which employs Range Image Projection to overcome the issues arising from the loss of information during 3D to 2D point cloud data conversion. It provides high performance on the SemanticKITTI data.

Instance Segmentation: The Instance (things) segmentation task deals not only with identifying the semantic class a pixel is associated with, but also the specific object instance to which it belongs. Fast R-CNN[20] employs an end-to-end trained RoI (Region of Interest) pooling layer to generate high-quality region proposals. Faster R-CNN [21] advanced this method by learning the attention mechanism with a Region Proposal Network (RPN). Faster R-CNN is flexible and robust to many follow-up improvements. While Faster R-CNN has two outputs for each candidate object, a class label, and a bounding-box offset, Mask R-CNN is the addition of a third

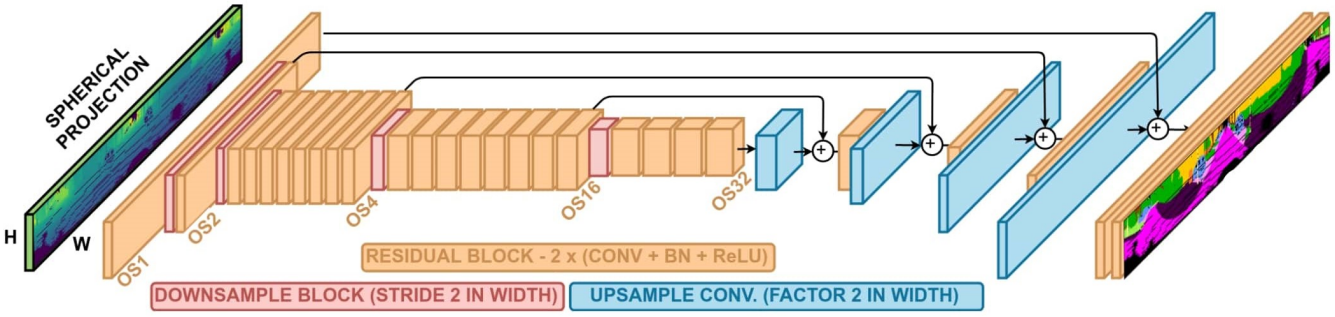

Fig. 1. RangeNet-based Common FPN backbone with cross connections

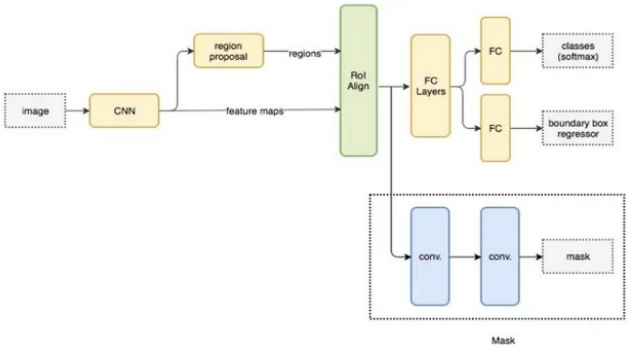

Fig. 2. Architecture of Instance Segmentation

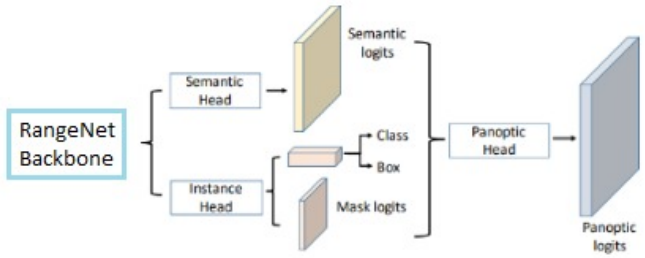

Fig. 3. Semantic, Instance and Panoptic Segmentation Heads

branch that outputs the object mask. This greatly simplifies the multi-stage the pipeline of original R-CNN. Thus, Mask R-CNN is the current leading framework in several benchmarks.

Panoptic Segmentation: The unified segmentation task for things and stuff classes has been studied over a long time, including early work on scene parsing [11], and image parsing [12]. Among the current state-of-the-art models, UPSNet[13] utilizes a single encoder-decoder network as a backbone to provide shared feature maps. It is designed primarily for COCO and Cityscapes datasets. Similar to UPSNet, EfficientLPS[14] uses a single network as a backbone. Going a step further, EfficientLPS effectively addresses the loss of information arising from 3D to the 2D conversion of LiDAR point cloud data by employing a 2D CNN for the task while explicitly utilizing the unique 3D information provided by point clouds.

Based on the literature review, given the current state-of-the-art performances for both models, we aim to implement the semantic head, the instance head based on Mask R-CNN, followed by a common panoptic head on the top of RangeNet backbone.

IV. Methodology

1. Dataset: Commonly used for perception-based tasks, Semantic KITTI collects 3-D point cloud LiDAR data using real-world inputs captured by Velodyne sensors. The dataset, which consists of many sequences recorded in various locations throughout the world, consists of Velodyne sensor data in BIN file format and ground truth labels for panoptic classes for each point-cloud data scan.

2. Point Cloud Input: LiDAR-based 3D point cloud data can be directly processed and utilized for semantic and instance segmentation to obtain class labels as well as instances for each point in the cloud. Another way to perform the same is by projecting the point cloud from 3D space to 2D space. This step reduces the dimensionality of the map. It is usually done either in spherical projection or in bird's eye view projection. Here, we use a spherical projection of the point cloud data as input to the backbone. A spherical projection (or image of a region) contains five channels (X, Y, Z, R, and I), each representing specific information about a 3D point. The X, Y, and Z channels correspond to the 3D point's X, Y, and Z coordinates. R is the point's Euclidean distance or extent from the origin and I is the point's intensity.

3. Encoder: The RangeNet encoder is based on YOLOv3's Darknet53 (53-layered). Batch normalization and LeakyReLU follow each one of the convolution laters. The encoder can be divided into numerous residual blocks, each one consisting of 2 convolution layers with 1x1 kernel size, and a kernel of size 3x3 follows it. The inputs to each of the residual blocks is added to different blocks in the decoder.

4. Decoder: The decoder employed in our model involves upsampling CNNs layers (to take the image back to its original dimensions) which are followed by a residual block and a dropout layer. Skipped links are included in the network and connect each encoder block input to the corresponding decoder

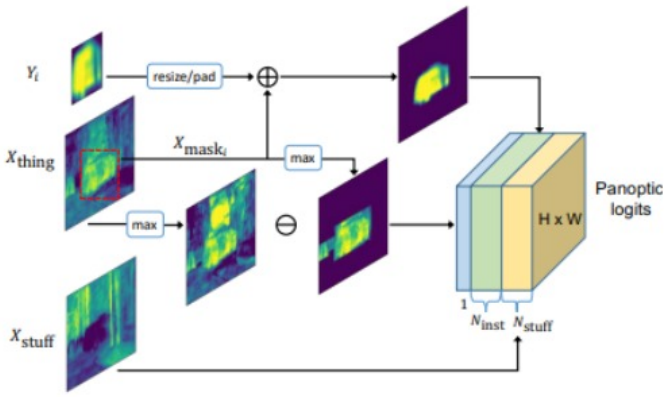

Fig. 4. Architecture of our Panoptic Head

block output. This helps us recover some of the high-frequency edge information lost during the downsampling process.

5. Semantic Segmentation Head: Following the decoder, the semantic segmentation involves a layer performing a 1 x 1 convolution to obtain the output feature maps of dimensions C x H x W where C is the number of classes. This is followed by a Softmax activation to obtain the pixel-wise probability distribution of the classes.

6. Instance Segmentation Head: The instance segmentation head consists of the following components - Region Proposal Network, Region Of Interest(RoI) Align Layer, a Softmax activation and a bounding box refinement function.

6.1. Region Proposal Network: The RPN runs over the image in a sliding-window manner and finds regions that contain objects. It uses anchor boxes to detect multiple objects, objects of different scales, and overlapping objects in an image. These are boxes distributed over the image area. There are about 100,000 anchors of various dimensions and aspect ratios that overlap to cover as much of the image as possible.

6.2. Bounding Box Refinement: A positive anchor (foreground anchor) might not be centered perfectly over the object. So the RPN estimates a change in x, y, width, and height to refine the anchor box to fit the object better.

6.3. RoIAlign Layer: The RoIAlign Layer is used to improve the spatial alignment of the feature maps from the decoder and the region proposals from the RPN Layer. The ROI Align Layer uses bilinear interpolation to warp the features in each ROI to the fixed size. This allows the features to be more accurately aligned to the objects in the input image which can improve the performance of the network.

6.4. Softmax activation: The output from RoIAlign Layer is passed through Fully-connected layers followed by a final softmax layer, to obtain the pixel-wise probability distributions.

7. Panoptic Segmentation Head: Once we have the semantic segmentation and instance segmentation results from the above two heads, we combine their outputs (specifically the logits per pixel) into the panoptic segmentation head. To begin, logits from semantic segmentation are divided into separate stuff and thing classes. Then stuff classes are directly taken in

$$\mathrm{PQ} = \underbrace{\frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP|}}_{\text{segmentation quality (SQ)}} \times \underbrace{\frac{|TP|}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}}_{\text{recognition quality (RQ)}}$$

$$\mathrm{PQ} = \frac{\sum_{(p,g)\in TP} \mathrm{IoU}(p,g)}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}$$

the panoptic head. For instance classes, mask matching is done by combining the logits from semantic and instance heads. After that, the mask removal technique is used to remove false positive predictions from the instance segmentation branch. Specifically, as our overall pipeline is in TensorFlow and UPSNet is written in PyTorch, we had to make a few changes and leave all the rest of the functionalities, to have a baseline Panoptic Head for our code. For Mask Matching, pixelwise, we have compared the masks recieved from semantic and instance heads, if it is having an high associativity we are then using the Mask Removal function to remove the redundant masks.

Training Process: Out of the vast (21 sequences) of data in the LiDAR Point Cloud Semantic KITTI data, the first seven were used for the training process, the next two for the validation, and the data 00 for testing purposes. The training set is then divided into batches depending on the batch size. Given the constraints of the hardware available, we could train with a batch size of 2 on NVIDIA A100-PCIE-40GB MIG 1g.5 GB, while the Google Colab Pro platform (P100 GPU) could train with a higher batch size of 4. The loss was calculated using the weighted categorical cross-entropy. The loss calculation did not include the unlabelled points and by making their weights equal to zero in the training process. The weights were tuned as the inverse square root of the class frequency.

Evaluation Metrics and Optimization: The evaluation metrics of LiDAR-based panoptic segmentation are the Panoptic Quality (PQ), Segmentation Quality (SQ), and Recognition Quality (RQ) which are calculated across all classes. The above three metrics are also calculated separately on things and stuff classes.

For each class, the unique matching splits the predicted and ground truth segments into three sets: true positives (TP), false positives (FP), and false negatives (FN), representing matched pairs of segments, unmatched predicted segments, and unmatched ground truth segments, respectively.

Mean IoU (mIoU) is also used to evaluate the quality of the sub-task of semantic segmentation. IoU is the ratio between correctly predicted pixels and the total number of pixels in either the prediction or ground truth for each class. I for semantic segmentation is distinct from our segmentation quality (SQ), which is computed as the average IoU over matched segments. We will train the networks on SemanticKitti and nuScenes

Architecture	Epochs	Batch Size	Training Time (min/epoch)	Accuracy	Cars	Pedestrian	Road	Parking	Sidewalk	Fence	Traffic Sign	Mean IoU
RangeNet (Base)	10	3	82	**68.96**	**84.72**	**0.26**	**94.84**	**58.21**	**82.2**	**28.39**	2.69	**50.15**
Our Architecture	10	3	89	61.73	82.18	0.21	89.67	41.02	73.89	23.01	**16.54**	45.81

TABLE I
QUANTITATIVE COMPARISON OF EVALUATION METRICS ON SEMANTIC-KITTI TEST DATASET

LiDAR point cloud dataset and evaluate the performance.

In addition to these, we have measured the precision and recall of the resultant model. Precision measures the proportion of true positive predictions made by the model, while recall measures the proportion of true positive examples that were correctly predicted by the model. Precision is defined as the number of true positive predictions made by the model divided by the total number of positive predictions made by the model. The recall is defined as the number of true positive predictions made by the model divided by the total number of true positive examples in the dataset. In general, a good model should have high precision and a high recall.

Attempts have been made to implement both SGD and Adam as optimizers. The final implementation will use the Adam optimizer for the following reasons:

1. Adam computes individual learning rates for various parameters. At each iteration, the actual step size obtained by Adam approximately limits the step size his hyperparameter.

2. The Adam update rule step size is independent of the gradient size. This is useful when traversing areas with small gradients (saddle points, canyons, etc.). In such regions, the stochastic gradient descent (SGD) optimizer has difficulty navigating them quickly.

V. EXPERIMENTS

The panoptic segmentation can be done in 3 ways in terms of Backbone. Usually, 2 separate backbones for semantic and instance segmentation are used. In the research developments, some architectures have implemented the network with a single encoder dual decoder. In this project, we have tried implementing the shared encoder-decoder for both the instance as well as semantic heads.

In the independent backbone architecture, we can parallel run instance and semantic segmentation in order to extract mask logits required for panoptic head processing in the form of tensors. Although there is a relative reduction in complexity(more than 65 percent), an accurate model is not guaranteed. Therefore, we tried taking this approach of shared encoder-decoder.

VI. RESULTS

Table 1 shows a quantitative comparison between the metrics Accuracy and IoU on the Semantic KITTI Dataset.

Baseline RangeNet showed better accuracy and mean IoU scores as compared to our modified architecture. But this result is due to a lack of proper model training and may require more training (up to 200 epochs) to get some conclusive results.

Also, since false negatives are more likely to lead to accidents than false positives, network implementations achieved reasonable IoU scores very close to RangeNet, especially in the desirable car class for autonomous driving. The low IoU values for the fence, pedestrian, and traffic sign classes are attributed to the very small number of instances of fences, pedestrians, and traffic signs in the dataset, and these fences, pedestrians, and traffic signs Signs instances are much smaller and have very fine detail, making segmentation difficult. It is clear that the roads and sidewalk classes have high IoU scores because these classes are very common in the dataset.

Figure 5 shows the output of the trained network.

VII. CHALLENGES

Through the background research that we performed, we were exposed to different computer vision techniques. However, Panoptic Segmentation is a recent development, and the documentation on it is currently limited. Moreover, Panoptic tasks involved both Semantic and Instance segmentation tasks to be performed simultaneously, which made it difficult. The major baselines that we had for reference were UPSNet, RangeNet, and Mask R-CNN developed over the past few years. Furthermore, these baselines worked primarily for a specific dataset (Eg. SemanticKITTI, COCO, Cityscapes). Thus, as RangeNet (utilized by us for semantic segmentation) was based on the SemanticKITTI dataset, we used the same dataset for our model. However, Mask R-CNN (which we were implementing for instance segmentation) was built on COCO, and thus, the task of making the SemanticKITTI data compatible with it is really complicated and caused numerous problems in the process. As neither RangeNet nor Mask R-CNN had panoptic heads, we referred to UPSNet. Again, UPSNet being based on Cityscapes data, it is really tough to make it compatible with SemanticKITTI. Thus, as our implementation is based on three different architectures, we had to analyze and adjust the feature pyramid shapes at multiple different locations for incorporating them. Furthermore, the UPSNet panoptic head employs multiple loss functions. Together with the dependencies from other files, there were eight loss functions, making it a really tough task to implement all of them to fit them into our model.

Also, the Velodyne-based SemanticKITTI dataset that we utilized was very large in size (80GB+) and in .bin format, so it took us a lot of time to explore it, set it up, and process it. Further, the data was in the original 3D Point Cloud format, whereas all the existing models take 2-D converted data as the input. After multiple trials, our devices were not able

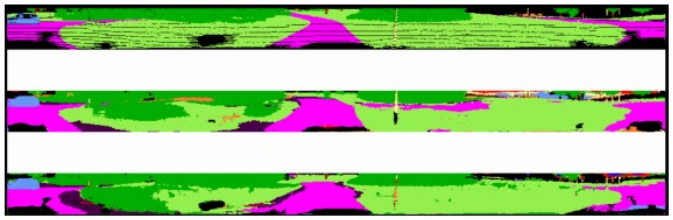

Fig. 5. 1) Top: Ground Truth 2) Middle: Output of our architecture 3) Bottom: Output from RangeNet

to implement it. The implementation required much better hardware than what was available to us. This made us shift to clusters, but due to a lack of computational resources, it again failed. Finally, we were able to run it to the semantic head. Further, we had to set up and implement CUDA modules on our devices. Further, as the dataset we have chosen to implement this architecture on, is rather large (80GB), this was one reason why it is taking a longer time to run. When we tried to run the Panoptic Architecture, we were unable to secure enough resources to run the entire architecture, as the WPI Cluster - Turing is a shared space and it won't allow us to use the resources for a longer time. Other Panoptic Architectures have run on Industrial GPUs (for eg; multiple Tesla V100s) and they achieved decent results only after 100 epochs, which would take a prolonged period to complete the run.

VIII. Conclusions and Future Work

In this project, we have explored the panoptic segmentation architectures and built one in TensorFlow 2.0 on the top of RangeNet backbone. It is observed that the architecture is complex in terms of computations as it is a combination of 3 different architectures semantic head, instance head, and panoptic head. Though considerable results are obtained with semantic segmentation, there are computational and complexity limitations with instance segmentation and panoptic segmentation.

For future work, we plan to reduce the complexity of our architecture and run the architecture with high computational hardware. Additionally, we would like to exploit our understanding of panoptic segmentation gathered from our analysis to make it with faster inference for real-time applications.

Acknowledgment

We would like to express our gratitude to Prof. Fabricio Murai for facilitating an intellectual dialogue on bolstering our basics and current state of the domain of Deep Learning.

References

[1] Xu Yan, Jiantao Gao, Chaoda Zheng, Chao Zheng, Ruimao Zhang, Shenghui Cui, Zhen Li. (2022). 2DPASS: 2D Priors Assisted Semantic Segmentation on LiDAR Point Clouds.
[2] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In CVPR, 2015
[3] R. Girshick. Fast R-CNN. In ICCV, 2015
[4] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In NIPS, 2015
[5] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask R- CNN. In ICCV, 2017
[6] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, Raquel Urtasun. (2019). UPSNet: A Unified Panoptic Segmentation Network. IEEE/CVF Conference
[7] Sirohi, K., Mohan, R., Büscher, D., Burgard, W., Valada, A. (2021). Efficientlps: Efficient lidar panoptic segmentation. IEEE Transactions on Robotics.
[8] Milioto, Andres, Vizzo, Ignacio, Behley, Jens, Stachniss, Cyrill. (2019). RangeNet ++: Fast and Accurate LiDAR Semantic Segmentation. 4213-4220. 10.1109/IROS40897.2019.8967762.
[9] Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick. (2018). Mask R-CNN. Facebook AI Research (FAIR)
[10] Alexander Kirillov, Ross Girshick, Kaiming He, Piotr Dollár. (2019). Panoptic Feature Pyramid Networks. Facebook AI Research (FAIR)
[11]] J. Tighe, M. Niethammer, and S. Lazebnik. Scene parsing with object instances and occlusion ordering. In CVPR, 2014.
[12] Z. Tu, X. Chen, A. L. Yuille, and S.-C. Zhu. Image parsing: Unifying segmentation, detection, and recognition. IJCV, 2005
[13] Yuwen Xiong, Renjie Liao, Hengshuang Zhao, Rui Hu, Min Bai, Ersin Yumer, Raquel Urtasun. (2019). UPSNet: A Unified Panoptic Segmentation Network. IEEE/CVF Conference
[14] Kshitij Sirohi, Rohit Mohan, Daniel Buscher, Wolfram Burgard, Abhinav Valada. EfficientLPS: Efficient LiDAR Panoptic Segmentation. IEEE/RSJ International conference (2021)
[15] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016)
[16] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, Juergen Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. IEEE/CVF International Conference on Computer Vision (ICCV) (2019)
[17] Bichen Wu, Alvin Wan, Xiangyu Yue, Kurt Keutzer. SqueezeSeg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud. IEEE International Conference on Robotics and Automation (ICRA) (2018)
[18] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, Kurt Keutzer. SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud. IEEE International Conference on Robotics and Automation (ICRA) (2019)
[19] Yang Zhang, Zixiang Zhou, Philip David, Xiangyu Yue, Zerong Xi, Boqing Gong, Hassan Foroosh. PolarNet: An Improved Grid Representation for Online LiDAR Point Clouds Semantic Segmentation. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2020)
[20] Ross Girshick. Fast R-CNN. IEEE International Conference on Computer Vision (ICCV) (2015)
[21] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence (2016)
[22] Jonathan Hui. Image segmentation with Mask R-CNN. Medium Digest (2018)
[23] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV, 2014