LU-Net: An Efficient Network for 3D LiDAR Point Cloud Semantic Segmentation Based on End-to-End-Learned 3D Features and U-Net

Pierre Biasutti[1,2,3,4], Vincent Lepetit[1], Mathieu Brédif[3], Jean-François Aujol[2], Aurélie Bugeau[1]

[1]*Univ. Bordeaux, CNRS, Bordeaux INP, LaBRI, UMR 5800, F-33400, Talence, France*
[2]*Univ. Bordeaux, IMB, INP, CNRS, UMR 5251, F-33400 Talence, France*
[3]*Univ. Paris-Est, LASTIG GEOVIS, IGN, ENSG, F-94160 Saint-Mandé, France*
[4]*GEOSAT, Pessac, France*
pierre.biasutti@labri.fr

Abstract

We propose LU-Net—for LiDAR U-Net, a new method for the semantic segmentation of a 3D LiDAR point cloud. Instead of applying some global 3D segmentation method such as PointNet, we propose an end-to-end architecture for LiDAR point cloud semantic segmentation that efficiently solves the problem as an image processing problem. We first extract high-level 3D features for each point given its 3D neighbors. Then, these features are projected into a 2D multichannel range-image by considering the topology of the sensor. Thanks to these learned features and this projection, we can finally perform the segmentation using a simple U-Net segmentation network, which performs very well while being very efficient. In this way, we can exploit both the 3D nature of the data and the specificity of the Li-DAR sensor. This approach outperforms the state-of-the-art by a large margin on the KITTI dataset, as our experiments show. Moreover, this approach operates at 24fps on a single GPU. This is above the acquisition rate of common LiDAR sensors which makes it suitable for real-time applications.

1. Introduction

The recent interest for autonomous systems has motivated many computer vision works over the past years. The importance of accurate perception models is a crucial step towards system automation, especially for mobile robots and autonomous driving. Modern systems are equipped with both optical cameras and 3D sensors, mostly LiDAR sensors. These sensors are now essential components of perception systems as they enable direct space measurements, providing an accurate 3D representation of the scene. However, for most automation-related tasks, raw Li-DAR point clouds require further processing in order to be
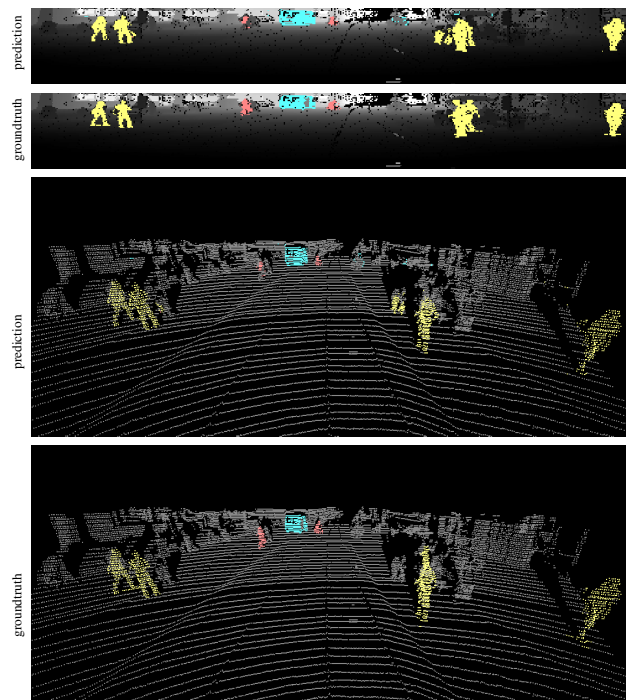

Figure 1. The top two images show the segmentation of LiDAR data obtained with our method, and the groundtruth segmentation, seen in the sensor topology. The bottom two images show the same segmentations from a different point of view.

used. In particular, point clouds with accurate semantic segmentation provide a higher level of representation of the scene that can be used in various applications such as obstacle avoiding, road inventory, or object manipulation.

This paper focuses on the semantic segmentation of 3D LiDAR point clouds. Given a point cloud acquired with a LiDAR sensor, we aim at estimating a label for each point that belongs to objects of interest in urban environments (such as cars, pedestrians and cyclists). The tradi-

tional pipelines used to tackle this problem consider ground removal, clustering of remaining structures, and classification based on handcrafted features extracted on each clusters [8, 6]. The segmentation can be improved with variational models [12]. These methods are often hard to tune as handcrafted features usually require tuning many parameters, which is likely to be data dependent and therefore hard to use in a general scenario. Finally, although the use of regularization can lead to visual and qualitative improvements, it often leads to a large increase of the computational time.

Recently, deep-learning approaches have been proposed to overcome the difficulty of tuning handcrafted features. This has become possible with the arrival of large 3D annotated datasets such as the KITTI 3D object detection dataset [7]. Many methods have been proposed to segment the point cloud by directly operating in 3D [17] or on a voxel-based representation of the point cloud [23]. However, this type of methods either needs very high computational power, or are not able to process the amount of points acquired in a single rotation of a sensor. Even more recently, faster approaches have been proposed [20, 19]. They rely on a 2D representation of the point cloud, called range-image [1], which can be used as the input of a convolutional neural network. Thus, the processing time as well as the required computational power can be kept low, as these range-images consist in low resolution, multichannel images. Unfortunately, the choice of input channels, as well as the difficulty of processing geo-spatial information using only 2D convolutions have limited the results of such approaches, which have not yet achieved good enough scores for practical use, especially on small objects classes such as cyclists or pedestrians.

In this paper, we propose LU-Net—for LiDAR U-Net—an end-to-end model for the semantic segmentation of 3D LiDAR point clouds. LU-Net benefits from a high-level 3D feature extraction module that can embed 3D local features in 2D range-images, which can later be efficiently used in a U-Net segmentation network. We demonstrate that, beside being a simple and efficient method, LU-Net largely outperforms state-of-the-art range-image methods, as shown in Figure 1.

The rest of the paper is organized as follows: We first discuss previous works on point cloud semantic segmentation, including methods designed for processing LiDAR data. We then detail our approach, and evaluate it on the KITTI dataset against state-of-the-art methods and discuss the results.

2. Related Work

In this section, we discuss previous works on image semantic segmentation as well as 3D point cloud semantic segmentation below.

2.1. Semantic Segmentation for Images

Semantic segmentation of images has been the subject of many works in the past years. Recently, deep learning methods have largely outperformed previous ones. The method presented in [16] was the first to propose an accurate end-to-end network for semantic segmentation. This method is based on an encoder in which each scale is used to compute the final segmentation. Only a few month later, the U-Net architecture [18] was proposed for the semantic segmentation of medical images. This method is an encoder-decoder able to provide highly precise segmentation. These two methods have largely influenced recent works such as DeeplabV3+ [5] that uses dilated convolutional layers and spatial pyramid pooling modules in an encoder-decoder structure to improve the quality of the prediction. Other approaches explore multi-scale architectures to produce and fuse segmentations performed at different scales [14, 22]. Most of these methods are able to produce very accurate results, on various types of images (medical, outdoor, indoor). The survey [3] of CNNs methods for semantic segmentation provides a deep analysis of some recent techniques. This work demonstrates that a combination of various components would most likely improve segmentation results on wider classes of objects.

2.2. Semantic Segmentation of Point Clouds

3D-based methods. As mentioned above, the first approaches for point cloud semantic segmentation were done using heavy pipelines, composed of many successive steps such as: ground removal, point cloud clustering, feature extraction as presented in [8, 6]. However, as mentioned above, these methods often require many parameters and they are therefore hard to tune. In [11], a deep-learning approach is used to extract features from the point cloud. Then, the segmentation is done using a variational regularization. Another approach presented in [17] proposes to directly input the raw 3D LiDAR point cloud to a network composed of a succession of fully-connected layers to classify or segment the point cloud. However, due to the heavy structure of this architecture, it is only suitable for small point clouds. Moreover, processing 3D data often increases the computational time due to the dimension of the data (number of points, number of voxels), and the absence of spatial correlation. To overcome these limitations, the methods presented in [13] and [23] propose to represent the point cloud as a voxel-grid which can be used as the input of a 3D CNN. These methods achieve satisfying results for 3D detection. However, semantic segmentation would require a voxel-grid of very high resolution, which would increase the computational cost as well as the memory usage.
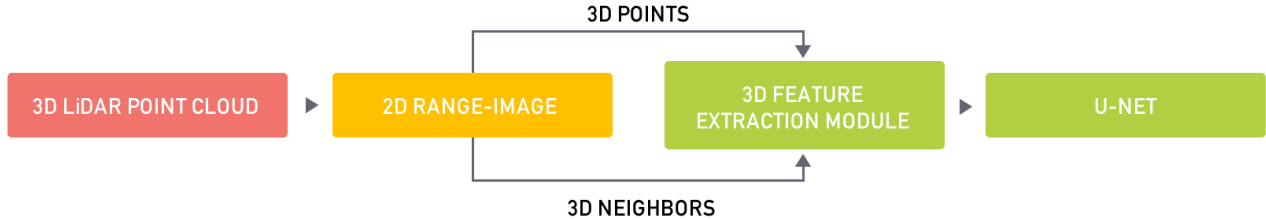
Figure 2. Proposed pipeline for 3D LiDAR point cloud semantic segmentation. First, the topology of the sensor is used to estimate the 8-connected neighborhood of each point. Then, each point and its neighbors are fed to the high-level 3D feature extraction module, which outputs a multichannel 2D range-image. The range-image is finally used as the input of a U-Net segmentation network.

Range-image based methods. Recently, SqueezeSeg, a novel approach for the semantic segmentation of a LiDAR point cloud represented as a spherical range-image [1], was proposed. This representation allows to perform the segmentation by using simple 2D convolutions, which lowers the computational cost while keeping good accuracy. The architecture is derived from the SqueezeNet image segmentation method [10]. The intermediate layers are "fire layers", *i.e.* layers made of one squeeze module and one expansion module. Later on, the same authors improved this method in [21] by adding a context aggregation module and by considering focal loss and batch normalization to improve the quality of the segmentation. A similar range-image approach was proposed in [19], where a Atrous Spatial Pyramid Pooling [4] and squeeze reweighting layer [9] are added. Finally, in [2], the authors offer to input a range-image directly to the U-Net architecture described in [18]. This method achieves results that are comparable to the state of the art of range-image methods with a much simpler and more intuitive architecture. All these range-image methods succeed in real-time computation. However, their results often lack of accuracy which limits their usage in real scenarios.

In the next section, we propose LU-Net: an end-to-end model for the accurate semantic segmentation of point clouds represented as range-images. We will show that it outperforms all other range-image methods by a large margin on the KITTI dataset, while offering a robust methodology for bridging between 3D LiDAR point cloud processing and 2D image processing.

3. Methodology

In this section, we present our end-to-end model for the semantic segmentation of LiDAR point clouds inspired by the U-Net architecture [18]. An overview of the proposed method is available in Figure 2.

3.1. Network input

As mentioned above, processing raw LiDAR point clouds is computationally expensive. Indeed, these 3D point clouds are stored as unorganized lists of (x, y, z) Cartesian coordinates. Therefore processing such data often involves preprocessing steps to bring spatial structure to the data. To that end, alternative representations, such as voxel grids or 2D pinhole projections in 2D images, are sometimes used, as discussed in the Related Work section. However, high resolution is often needed in order to represent enough details, which involves heavy memory costs. Modern LiDAR sensors often acquire 3D points, following a strict sensor topology, from which we can build a dense 2D image [1], the so-called range-image. The range-image offers a lightweight, structured and dense representation of the point cloud.

3.2. Range-images

Whenever the raw LiDAR data (with beam number) is not available, the point cloud has to be processed to extract the corresponding range-image. As 3D LiDAR sensors acquire 3D points with a sampling pattern of a few number of scan lines and quasi uniform angular steps between samples, the acquisition follows a grid pattern that can be used to create a 2D image. Indeed, each point is defined by two angles and a depth, (θ, ϕ, d) respectively, with steps of $(\Delta\theta, \Delta\phi)$ between two consecutive positions. Each point p_i of the LiDAR point cloud P can be mapped to the coordinates (x, y) with $x = \lfloor \frac{\theta}{\Delta\theta} \rfloor, y = \lfloor \frac{\phi}{\Delta\phi} \rfloor$ of a 2D range-image u of resolution $H \times W = Card(P)$, where each channel represents a modality of the measured point. A range-image is presented on Figure 3.

In perfect conditions, the resulting image is completely dense, without any missing data. However, due to the nature of the acquisition, some measurements are considered invalid by the sensor and they lead to empty pixels (no-data). This happens when the laser beam is highly deviated (*e.g.* when going through a transparent material) or when it does not create any echo (*e.g.* when the beam points in the sky direction). We propose to identify such pixels using a binary mask m equal to 0 for empty pixels and to 1 otherwise. The analysis of multi-echo LiDAR scans is subject to future work.

3.3. High-level 3D feature extraction module

In [19], [20] and [21], the authors use a 5-channel range-image as input of their network. These 5 channels are made of the 3D coordinates (x, y, z), the reflectance (r) and the spherical depth (d). However, the analysis presented in [2] showed that feeding a 2-channel range-image with only the reflectance and depth information to a U-Net architecture achieves comparable results to the state of the art.

In all these previous works, the choice of the number of channels of the range-image appears to be empirical. For each application, a complete study or a large set of experiments must be conducted to choose the best within all the possible combinations of channels. This is tedious and time consuming. To bypass such an expensive study, we propose in this paper a feature extraction module that is able to directly learn meaningful features adapted to the target application—here, semantic segmentation.

Moreover, processing geo-spatial information using 2D convolutional layers can cause issues in terms of data normalization as LiDAR sensors sampling typically decreases when acquiring farther points.

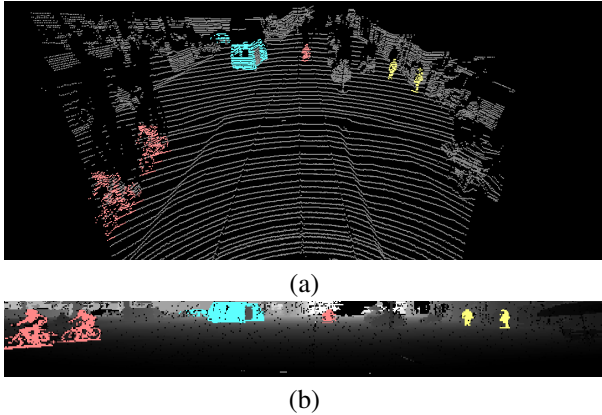Inspired by the Local Point Embedder presented in [11],

(a)

(b)

Figure 3. Turning a point cloud into a range-image. (a) A point cloud from the KITTI database [7], (b) the same point cloud as a a range-image. Note that the dark area in (b) corresponds to pulses with no returns. Colors correspond to groundtruth annotation, for better understanding.
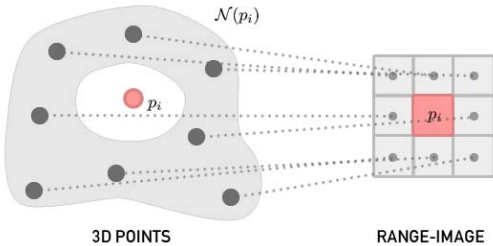

Figure 4. Illustration of the notation of the input of the feature extraction module. p_i is the point, $\mathcal{N}(p_i)$ is the set of neighbors of p_i.
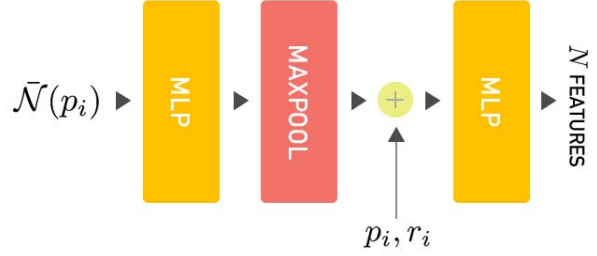

Figure 5. Architecture of the 3D feature extraction module. The output is an $1 \times N$ feature vector for each LiDAR point.

we propose a high-level 3D feature extraction module that is able to learn N meaningful high-level 3D features for each point and to output a range-image with N channels. Contrary to [11], our module exploits the range-image to directly estimate the neighbors of each points instead of using a pre-processing step. Moreover, our module outputs a range-image, instead of a point cloud, which can be used as input to a CNN.

Given a point $p_i = (x, y, z)$, and r_i its associated reflectance, we define $\mathcal{N}(p_i)$ the set of neighboring points of p_i in the range-image (*e.g.* the points that correspond to the 8-connected neighborhood of p_i in the range-image). This set is illustrated Figure 4. We also define $\bar{\mathcal{N}}(p_i) = \{q - p_i \mid q \in \mathcal{N}(p_i)\}$ the set of neighbors in coordinates relative to p_i. Note that if either p_i or q is an empty pixel, then $q - p_i = (0, 0, 0)$.

Similarly to [11], the set of neighbors $\bar{\mathcal{N}}(p_i)$ is first processed by a multi-layer perceptron (MLP), which consists of a succession of linear, ReLU and batch normalization layers. The resulting set is then maxpooled to a point feature set, which is concatenated with p_i and r_i. The resulting vector is processed through another MLP that outputs a vector of N 3D features for each p_i. This module is illustrated in Figure 5.

As linear layers can be done using 1×1 convolutional layers, the whole P point cloud can be processed at once. In this case, the output of the 3D feature extraction module is a $\text{Card}(P) \times N$ matrix, which can then be reshaped to a $H \times W \times N$ range-image.

3.4. Semantic segmentation

Architecture. The U-Net architecture [18] is an encoder-decoder. As illustrated in Figure 6, the first half consists in the repeated application of two 3×3 convolutions followed by a rectified linear unit (ReLU) and a 2×2 max-pooling layer that downsamples the input by a factor 2. Each time a downsampling is done, the number of features is doubled to compensate for the loss of resolution. The second half of the network consists of upsampling blocks where the input is upsampled using 2×2 up-convolutions. Then, concatenation is done between the upsampled feature map and the
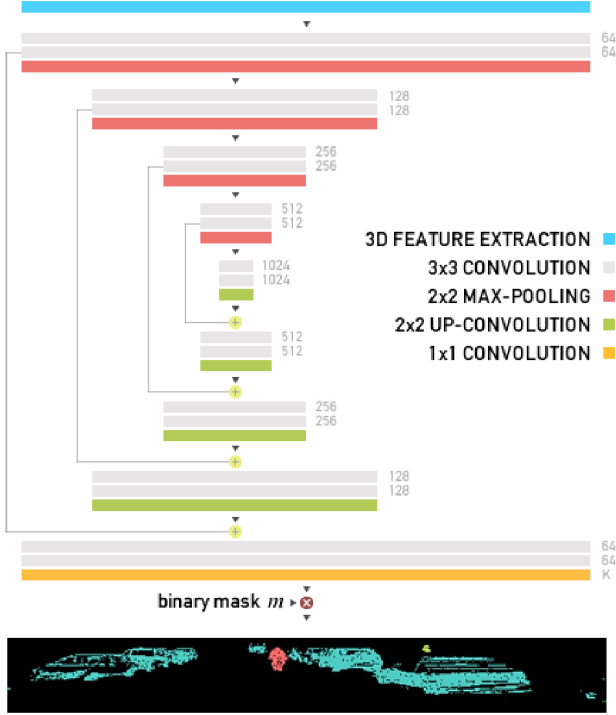
Figure 6. LU-Net architecture with the output of the 3D feature extraction module as the input (top) and the output segmented range-image (bottom).

corresponding feature map of the first half. This allows the network to capture global details while keeping fine details. After that, two 3×3 convolutions are applied followed by a ReLU. This block is repeated until the output of the network matches the dimension of the input. Finally, the last layer consists in a 1x1 convolution that outputs as many features as the wanted number of possible labels *i.e.* K 1-hot encoded.

Loss function. The loss function of our model is defined as a variation of the focal loss presented in [15]. Indeed, our model is trained on a dataset in which the number of example for each class is largely unbalanced. Using the focal loss approach helps improving the average score by few percents, as discussed later in Section 4. First, we define the pixel-wise softmax for each label k:

$$p_k(x) = \frac{\exp(a_k(x))}{\sum\limits_{k'=0}^{K} \exp(a_{k'}(x))}$$

where $a_k(x)$ is the activation for feature k at the pixel position x. After that, we define $l(x)$ the groundtruth label of pixel x. We then compute the weighted focal loss as follows:

$$E = \sum_{x \in \Omega} -\mathbb{1}_{\{m(x)>0\}} w(x)(1 - p_{l(x)}(x))^{\gamma} \log(p_{l(x)}(x))$$

where Ω is the domain of definition of u, $m(x) > 0$ are the valid pixels, $\gamma = 2$ is the *focusing* parameter and $w(x)$ is a weighting function introduced to give more importance to pixels that are close to a separation between two labels, as defined in [18].

Training We train the network with the Adam stochastic gradient optimizer and a learning rate set to 0.001. We also use batch normalization with a momentum of 0.99 to ensure good convergence of the model. Finally, the batch size is set to 4 and the training is stopped after 10 epochs.

4. Experiments

We trained and evaluated LU-Net using the same experimental setup as the one presented in SqueezeSeg [20] as they provide range-images with segmentation labels exported from the 3D object detection challenge of the KITTI dataset [7]. They also provide the training / validation split that they used for their experiments, which contains 8057 samples for training and 2791 for validation and which can be used for a fair comparison between each result of each method.

We have manually tuned the number of layers N, *i.e.* the number of 3D features learned for each points. On all our experiments, best semantic segmentation results were obtained by setting $N = 3$. This small amount of channels is enough to highlight the structure of the objects that are latter used in the U-Net in charge of the segmentation task. All results reported in this section are with this value. Nevertheless, if using the high-level 3D feature extraction module for other applications, one should consider adapting this value.

4.1. Comparison with the state of the art

We compare the proposed method to 4 range-image based methods of the state of the art: PointSeg [19], SqueezeSeg [20], SqueezeSegV2 [21], and RIU-Net [2]. RIU-Net is a previous version of LU-Net we developed and was solely based on the raw reflectance and depth features instead of the 3D features learned in the end-to-end network of LU-Net. Similarly to [20] and [21], the comparison is done based on the Intersection-over-Union score:

$$\text{IoU}_l = \frac{|\rho_l \bigcap G_l|}{|\rho_l \bigcup G_l|}$$

where ρ_l and G_l denote the predicted and groundtruth sets of points that belongs to label l respectively.

The performance comparisons between LU-Net and state-of-the-art methods are displayed Table 1. The first observation is that the proposed model outperforms existing

methods in terms of average IoU by over 10%. In particular, the proposed model achieves better results on each of the classes compared to PointSeg, SqueezeSeg and RIU-Net. Our method also largely outperforms SqueezeSegV2 for both pedestrians and cyclists.

Our method is very similar to RIU-Net as both methods use a U-Net architecture with a range-image as input. While RIU-Net uses 2 channels—the reflectance and depth—LU-Net automatically extracts a N-dimensional high-level features per point thanks to the 3D feature extraction module. Table 1 demonstrates that using an additional network to automatically learn high-level features from the 3D point cloud largely improves the results, especially on classes that are less represented in the dataset.

Figure 7 presents visual results for SqueezeSegV2 and LU-Net. We here observe that visually, the results for cars are comparable. Nevertheless, by looking closer at the results, we observe that SqueezeSegV2 is more subject to false positives (Figure 7, orange rectangle). Moreover, our method provides a better segmentation of the cars in the back of the scene, compared to SqueezeSegV2 (Figure 7, purple rectangle).

4.2. Ablation study

Table 2 presents intermediate scores in order to highlight the contribution of some model components.

First, we analyse the influence of relative coordinates $\bar{\mathcal{N}}$ as input to the 3D feature extraction module (Figure 5). We trained and tested the model using absolute coordinates $\mathcal{N}$. We name this version LU-Net w/o relative. As Table 2 shows, relative candidates provide better results than neighbors in absolute coordinates. We believe that by reading relative coordinates as input, the network learns high-level features characterizing the local 3D geometry of the point cloud, independently of its absolute position in the 3D environment. These absolute positions are re-introduced once this geometry is learned, *i.e.* before the second multi-layer

Ground truth

SqueezeSegV2 [21]

LU-Net

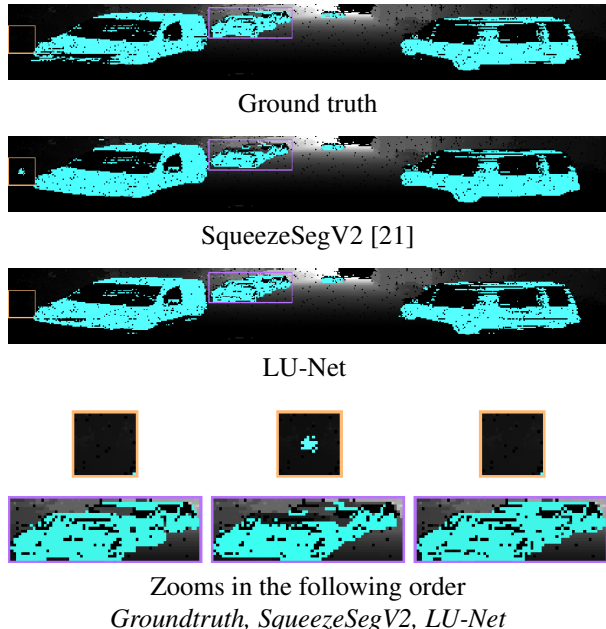Zooms in the following order
Groundtruth, SqueezeSegV2, LU-Net

Figure 7. Visual comparison of the proposed model against SqueezeSegV2 [21] and the ground truth. Results are shown on the range-image where depth values are encoded with a grayscale map. Both SqueezeSegV2 and LU-Net globally achieve very satisfying results. Nervertheless, LU-Net is less subject to false positives than SqueezeSegV2, as can be seen in the orange areas and corresponding zooms. It also better segments farther objects such as the cars on the back of the scene in the purple rectangle, which reduces the amount of false-negatives, which are crucial for autonomous driving applications.

Groundtruth

LU-Net w/o relative
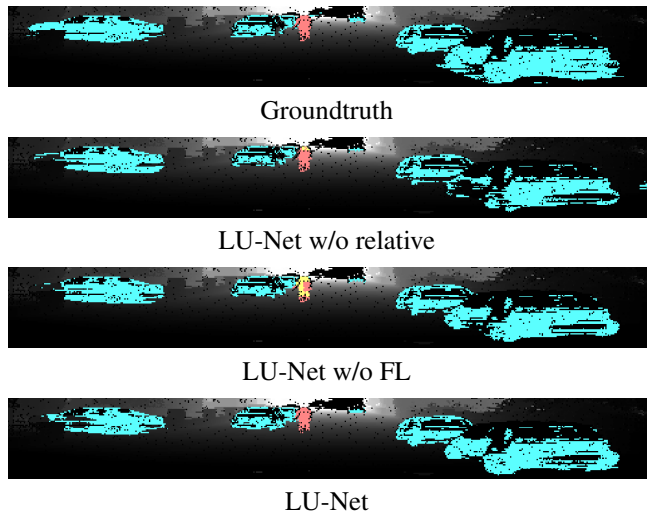
LU-Net w/o FL

LU-Net

Figure 8. Visual results of the ablation study. The use of neighbors in absolute coordinates results in incomplete segmentations of the objects compared to neighbors in relative coordinates. Moreover, the use of the focal loss (FL) helps the network to better distinguish classes that have similar aspects, here, cyclists and pedestrians.

perceptron of the 3D feature extraction module.

Table 1. Comparison (IoUs, %) of LU-Net with the state of the art for the semantic segmentation of the KITTI dataset.

		Cars	Pedestrians	Cyclists	Average
SqueezeSeg	[20]	64.6	21.8	25.1	37.2
PointSeg	[19]	67.4	19.2	32.7	39.8
RIU-Net	[2]	62.5	22.5	36.8	40.6
SqueezeSegv2	[21]	73.2	27.8	33.6	44.9
LU-Net		72.7	**46.9**	**46.5**	**55.4**

Table 2. Ablation study for the semantic segmentation of the KITTI dataset. Results in terms of (IoUs, %) for LU-Net w/o relative: which uses absolute coordinates $\mathcal{N}$ instead of relative $\bar{\mathcal{N}}$ as input to the feature extraction module; LU-Net w/o FL : proposed model without focal-loss; LU-Net: proposed model with relative coordinates and focal-loss.

	Cars	Pedestrians	Cyclists	Average
LU-Net w/o relative	62.8	39.6	37.5	46.6
LU-Net w/o FL	**73.8**	42.7	32.9	49.8
LU-Net	72.7	**46.9**	**46.5**	**55.4**

For fair comparison, we also experimented using absolute coordinates $\mathcal{N}$ and adding a supplementary convolutional layer as the first layer. Indeed, we could expect this additional layer to characterize the transformation from absolute and local coordinates. Nevertheless, this architecture brought numerical instability while not managing to learn such transform, as it ended up with an average IoU of 30.6%.

Next, we analyse the influence of the focal-loss. As seen in Table 2, the use of focal-loss largely improves the scores on both cyclists and pedestrians. This is related to the imbalance between each class in the dataset, where there are 10 times more car examples than cyclists or pedestrians.

4.3. Additional results

Apart from being convincing in terms of IoUs, the results produced by our method are also very convincing visually, as it is demonstrated Figure 1 and 9. Our segmentations are very close to those of the groundtruth. In Figure 9d), one of the pedestrians was not detected. When looking closely at the depth values in the range-image, this pedestrian is in fact hardly visible. It is also the case in the reflectance image. This is also related to the resolution of the sensor as only few points fall on the pedestrian, and could probably be solved by adding an external modality such as an optical image.

In Figure 9e), a car in the foreground is missing from the groundtruth, this causes the IoU to drop from 89.7% when ignoring this region of the image, down to 36.4%. Thus, removing examples with wrong or missing annotations in the dataset could lead to better results on LU-Net as well as on other methods. However, due to the amount of examples in the dataset, having a perfect annotation is practically very difficult.

Finally, LU-Net is able to operate at 24 frames per second on a single GPU. This is a lower frequency compared to other systems, yet still above the frame rate of the LiDAR sensor (10fps for the Velodyne HDL-64e). Moreover, our system uses only a few more parameters than RIU-Net for a significant improvement in terms of IoU scores.

5. Conclusion

In this paper, we have presented LU-Net, an end-to-end model for the semantic segmentation of 3D LiDAR point clouds. Our method efficiently creates a multi-channel range-image using a learned 3D feature module. This range-image later serves as the input of a U-Net architecture. We show that this methodology efficiently bridges between 3D point cloud processing and image processing. The resulting method is simple, but yet provides very high quality results far beyond existing state-of-the-art methods.

The current method relies on the focal loss function. We plan to study possible spatial regularization schemes within this loss function. Finally, fusion of LiDAR and optical data would probably enable reaching a higher level of accuracy.

6. Acknowledgement

References

[1] P. Biasutti, J.-F. Aujol, M. Brédif, and A. Bugeau. Range-Image: Incorporating Sensor Topology for Li-DAR Point Cloud Processing. *Photogrammetric Engineering & Remote Sensing*, 84(6):367–375, 2018.

[2] P. Biasutti, A. Bugeau, J-F. Aujol, and A. Brédif. RIU-Net: Embarrassingly simple semantic segmentation of 3D LiDAR point cloud. *arXiv Preprint*, 2019.

[3] A. Briot, P. Viswanath, and S. Yogamani. Analysis of Efficient CNN Design Techniques for Semantic Segmentation. In *Conference on Computer Vision and Pattern Recognition*, pages 663–672, 2018.

[4] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018.

[5] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and A. Hartwig. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *European Conference on Computer Vision*, pages 801–808, 2018.
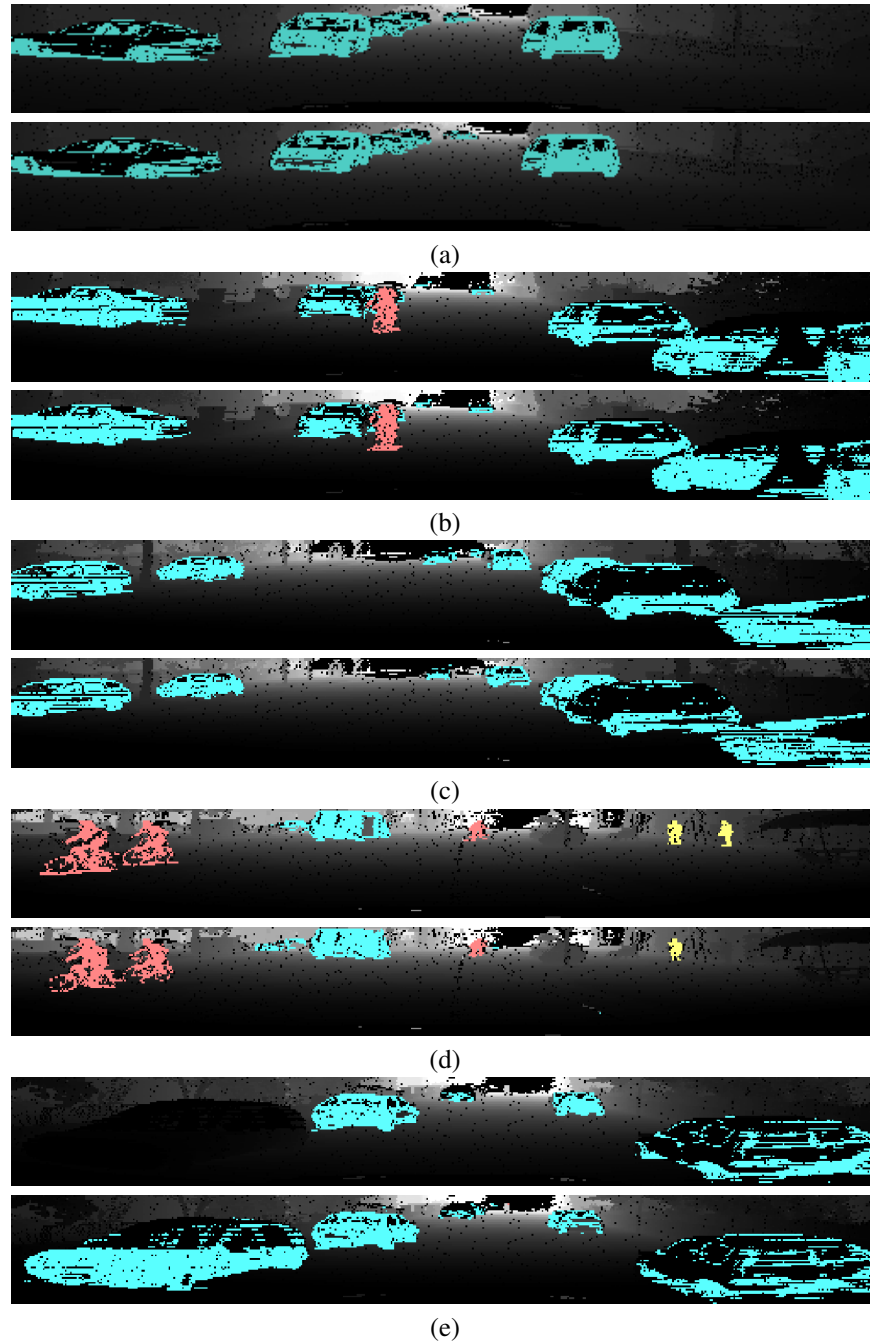
Figure 9. Results of the semantic segmentation of the proposed method (bottom) and groundtruth (top). Results are shown on the range-image where depth values are encoded with a grayscale map. Labels are associated to colors as follows: blue for cars, red for cyclists and lime for pedestrians.

[6] C. Feng, Y. Taguchi, and V. R. Kamat. Fast Plane Extraction in Organized Point Clouds Using Agglomerative Hierarchical Clustering. In *International Conference on Robotics and Automation*, pages 6218–6225, 2014.

[7] A. Geiger, P. Lenz, and R. Urtasun. Are We Ready for Autonomous Driving? the KITTI Vision Benchmark Suite. In *Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.

[8] M. Himmelsbach, A. Mueller, T. Lüttel, and H.-J. Wünsche. Lidar-Based 3D Object Perception. In *Proceedings of international workshop on Cognition for Technical Systems*, pages 1–7, 2008.

[9] J. Hu, L. Shen, and G. Sun. Squeeze-And-Excitation Networks. In *Conference on Computer Vision and Pattern Recognition*, pages 7132–7141, 2018.

[10] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer. Squeezenet: AlexNet-Level Accuracy with 50x Fewer Parameters and < 0.5 MB Model Size. In *arXiv Preprint*, 2016.

[11] L. Landrieu and M. Boussaha. Point Cloud Oversegmentation with Graph-Structured Deep Metric Learning. In *Conference on Computer Vision and Pattern Recognition*, 2019.

[12] L. Landrieu and M. Simonovsky. Large-Scale Point Cloud Semantic Segmentation with Superpoint Graphs. In *Conference on Computer Vision and Pattern Recognition*, pages 4558–4567, 2018.

[13] B. Li. 3D Fully Convolutional Network for Vehicle Detection in Point Cloud. In *International Conference on Intelligent Robots and Systems*, pages 1513–1518, 2017.

[14] G. Lin, A. Milan, C. Shen, and I. Reid. RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation. In *Conference on Computer Vision and Pattern Recognition*, pages 1925–1934, 2017.

[15] T-Y. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *PAMI*, 2018.

[16] J. Long, E. Shelhamer, and T. Darrell. Fully Convolutional Networks for Semantic Segmentation. In *Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[17] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[18] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *MICCAI International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015.

[19] Y. Wang, T. Shi, P. Yun, L. Tai, and M. Liu. Pointseg: Real-Time Semantic Segmentation Based on 3D LiDAR Point Cloud. In *arXiv Preprint*, 2018.

[20] B. Wu, A. Wan, X. Yue, and K. Keutzer. Squeezeseg: Convolutional Neural Nets with Recurrent CRF for Real-Time Road-Object Segmentation from 3D LiDAR Point Cloud. In *International Conference on Robotics and Automation*, pages 1887–1893, 2018.

[21] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer. SqueezesegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud. In *International Conference on Robotics and Automation*, pages 4376–4382, 2018.

[22] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. ICNet for Real-Time Semantic Segmentation on High-Resolution Images. In *European Conference on Computer Vision*, pages 405–420, 2018.

[23] Y. Zhou and O. Tuzel. VoxelNet: End-To-End Learning for Point Cloud Based 3D Object Detection. In *Conference on Computer Vision and Pattern Recognition*, pages 4490–4499, 2018.