Semi-Local Convolutions for LiDAR Scan Processing

Anonymous Author(s)
Affiliation
Address
email

Abstract

A number of applications, such as mobile robots or automated vehicles, use LiDAR sensors to obtain detailed information about their three-dimensional surroundings. Many methods use image-like projections to efficiently process these LiDAR measurements and use deep convolutional neural networks to predict semantic classes for each point in the scan. The spatial stationary assumption enables the usage of convolutions. However, LiDAR scans exhibit large differences in appearance over the vertical axis. Therefore, we propose semi local convolution (SLC), a convolution layer with reduced amount of weight-sharing along the vertical dimension. Our experiments did not show any improvement over traditional convolution layers in terms of segmentation IoU or accuracy.

1 Introduction

For a wide variety of applications it is important to understand the semantic meaning of the three dimensional world. LiDAR sensors provide a precise sampling of the environment and are often used for object detection and semantic segmentation. Many state-of-the-art semantic segmentation approaches make use of traditional two-dimensional CNNs by projecting the point clouds into an image-like structure [8, 4].

In this paper we provide deeper insights into one of the methods developed in [7]. We investigate whether the spatial stationary assumption of convolutions is still applicable to inputs with varying statistical properties over parts of the data, such as projected LiDAR scans. These data structures exhibit similar features as aligned images for which locally connected layers have been introduced [6]. Fig. 1 shows an example of such a projected LiDAR scan, which has a high variance in depth over the vertical axis, but not the horizontal axis. Further, the reflecivity measurements show inconsistencies over the vertical axis. To address this effect, we introduce SLC, a layer with reduced weight-sharing along the vertical spatial dimension.

2 Related Work

Convolution layers apply a filter bank on their input. The filter weights are shared over all spatial dimensions, meaning that for every location in the feature map the same set of filters are learned. The re-usability of weights causes a significant reduction in the number of parameters compared to fully connected layers. This allows deep convolutional neural networks to be trained successfully, in turn leading to a substantial performance boost in many computer vision applications. The underlying premise of convolutional methods is that of translational symmetry, i.e. that features that have been learned in one region of the image are useful in other regions as well.

For applications such as face recognition which deal with aligned data, locally connected layers have proved to be advantageous [2, 3, 6]. These layers also apply a filter bank. Contrary to convolutional

Under review at ICBINB Workshop @ NeurIPS 2021. Do not distribute.

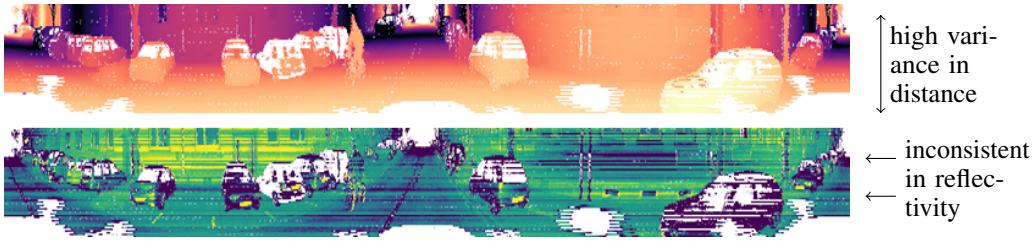

Figure 1: **LiDAR Scan**: A projected LiDAR scan has a high variance in distance over the vertical axis of the scan (top) and some sensors have calibration issues of the reflectivity module that varies for each layer (bottom).

layers, weights are not shared among the different locations in the feature map, allowing different sets of filters to be learned for every location in the input.

The spatial stationary assumption of convolutions does not hold for aligned images due to different local statistics in distant regions of the image. In a projected LiDAR scan, the argument holds true for sensors that are mounted horizontally. Each horizontal layer is fixed at a certain vertical angle. As the environment of the sensor is not invariant against rotations around this axis, this leads to different distance statistics in each vertical layer.

3 Method

We consider an input feature map x of shape $[H_x, W_x, C_x]$, representing a cylindrical projection with height H_x, width W_x, and channels C_x. It is passed through a convolution layer which outputs another feature map y of shape $[H_y, W_y, C_y]$.

A conventional 2D convolution layer computes the output y with a cross-correlation of the input x and a filter kernel k such that

$$y_{h,w,c_y} = \sum_{c_x=0}^{C_x-1} \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} k_{i,j,c_x,c_y} \cdot x_{h+i-I,w+j-J,c_x} \tag{1}$$

where k has the shape $[2I+1, 2J+1, C_x, C_y]$.

In a SLC layer, the filter kernel has the shape $[2I+1, 2J+1, C_x, C_y, \alpha]$ where α is the number of components with $\{\alpha \in \mathbb{N} : 1 \leq \alpha \leq H_x\}$. Each component is responsible for different parts along the vertical axis of the input (note that this concept can also be applied to the horizontal direction). The output of the SLC is then given by

$$y_{h,w,c_y} = \sum_{c_x=0}^{C_x-1} \sum_{i=0}^{I-1} \sum_{j=0}^{J-1} k_{i,j,c_x,c_y,\alpha_h} \cdot x_{h-i_I,w-j-J,c_x} \tag{2}$$

where $\alpha_h = \lfloor \alpha \cdot h / H_x \rfloor$ selects the respective filter-component depending on the vertical position h.

For $\alpha = H_x$, there is no weight sharing along the vertical axis, a new filter is used for every single data row. For $\alpha = 1$, we obtain a regular convolution as defined in Eq. (1). For values in between, the degree of weight sharing can be adapted to the desired application. All equations omit padding, bias, and activation function for better readability.

4 Experiments

In this section, we investigate the introduction of SLC layers in various experiments. We use the implementation of Milioto et al. [4][1] for our experiments. All results are reported on the validation split of the SemanticKITTI [1] dataset. The input to the network is a two-channel image with the projected depth measurements and the respective reflecivity values.

[1]Code: https://github.com/PRBonn/lidar-bonnetal

Table 1: **Overall Results**: Shown is the semantic segmentation performance over 19 object classes for two different base networks on the validation split of the SemanticKITTI dataset. Each base network is augmented with our SLC layer and tested for different values of α, i.e. the number of vertical filters within the convolution. All convolution layers, except input and output layer, are replaced within the network.

Network	Metric	# vertical filters α						
		1	2	4	8	16	32	64
DarkNet21 [4]	Accuracy	**0.83**	0.82	0.81	0.79	0.75	0.73	-
	mIoU	**0.36**	0.34	0.32	0.30	0.27	0.26	-
SqueezeSegV2 [8]	Accuracy	**0.84**	0.82	0.81	0.80	0.76	0.75	0.71
	mIoU	**0.36**	0.35	0.31	0.30	0.27	0.26	0.25

Table 2: **Scaled Network Performance**: This table shows the semantic segmentation performance of the DarkNet21 [4] model (Accuracy / mIoU). Over the columns, we increase the number of vertical filters within each SLC layer. Over the rows, we decrease the number of overall output filters for each layer, i.e. 2 means multiplying the number of filters by a factor of $\frac{1}{2}$ which results in $\frac{1}{4}$ of the original trainable variables. The gray cells mark those that contain configurations where the increase in α is neutralized with the decrease of filter sizes and thus results in approximately the same number of trainable parameters.

	1	2	4	8	16	32	64
1	0.83 / 0.36	0.82 / 0.34	0.81 / 0.32	0.79 / 0.30	0.75 / 0.27	0.73 / 0.26	-
2	0.83 / 0.35	0.81 / 0.32	0.83 / 0.33	- / -	- / -	- / -	0.70 / 0.24
4	0.82 / 0.34	- / -	0.77 / 0.30	0.76 / 0.28	0.74 / 0.25	- / -	0.72 / 0.24
8	0.77 / 0.31	- / -	- / -	0.74 / 0.27	0.72 / 0.25	0.71 / 0.24	0.71 / 0.23

Table 1 shows the overall results for two different base networks. First, it shows that SLCs do not outperform normal convolutions. Second, the performance decreases with increasing α. Third, the above two points apply to both base networks, even though DarkNet21 has approximately 24.7M trainable parameters, whereas SqueezeSegV2 only has approximately 928.5k parameters.

When α increases with a factor of 2, then also the number of trainable parameters increases by a factor of 2 (approximately).[2] Therefore, we conduct a second row of experiments, where we reduce the number of trainable parameters in the base network by decreasing the number of filters in each layer. Naturally, we expect a network with lower capacity to perform worse. Table 2 shows the semantic segmentation accuracy and mIoU performance for the DarkNet21 model. The entries with the gray background mark those networks that have approximately the same number of trainable variables as the base network, since the modification in α and output filter size cancel each other out. Here again, an increase in α leads to decreased performance. One has to note, that even if the number of parameters is the same for the gray cells, the increase in α only leads to more capacity over the spatial dimension of the feature maps, whereas larger output filters in general lead to more capacity over the depth of the network for the entire spatial extent.

5 Discussion

The experiments show that SLCs are not able to outperform normal convolutions and performance usually decreases with increasing α. This effect is stronger for networks with a large number of parameters. Therefore, we assume that normal convolution layers of adequate capacity can already handle the different statistical properties across the vertical spatial dimension. This claim is supported by the findings of Kayhan et al. [5] who show that convolutional layers exploit absolute spatial location. Therefore CNNs are in fact not translation invariant which means a network with sufficient capacity is able to learn even differing statistics over the vertical dimension of such a LiDAR scan.

[2] DarkNet21 with $\alpha = 64$ is too large for a single GPU, therefore no results are reported

6 Conclusion

This paper presented semi local convolutions, a network layer that only has limited weight sharing along the vertical spatial dimension of the input. Our experiments showed that using SLCs instead of normal convolutions decreases segmentation performance, especially when decreasing the amount of weight sharing. Since normal convolution layers can already exploit spatial location information within the network, it is not necessary to explicitly address the large difference in appearance along the vertical axis of a LiDAR scan in a special layer.

References

[1] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*, 2019.

[2] Karo Gregor and Yann LeCun. Emergence of complex-like cells in a temporal product network with local receptive fields. *arXiv.org*, 2010.

[3] G.B. Huang, Honglak Lee, and Erik Learned-Miller. Learning hierarchical representations for face verification with convolutional deep belief networks. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2518–2525, 06 2012.

[4] Andres Milioto, Ignacio Vizzo, Jens Behley, and Cyrill Stachniss. RangeNet ++: Fast and Accurate LiDAR Semantic Segmentation. In *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*, pages 4213–4220, 2019.

[5] Osman Semih Kayhan and Jan C. van Gemert. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 14262–14273, 2020.

[6] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1701–1708, 2014.

[7] Larissa T. Triess, David Peter, Christoph B. Rist, and J. Marius Zöllner. Scan-based Semantic Segmentation of LiDAR Point Clouds: An Experimental Study. In *Proc. IEEE Intelligent Vehicles Symposium (IV)*, pages 1116–1121, 2020.

[8] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud. In *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*, pages 4376–4382, 2019.