Dilated Point Convolutions: On the Receptive Field Size of Point Convolutions on 3D Point Clouds

Francis Engelmann[1], Theodora Kontogianni[1], Bastian Leibe[1]

***Abstract*— In this work, we propose *Dilated Point Convolutions* (DPC). In a thorough ablation study, we show that the receptive field size is directly related to the performance of 3D point cloud processing tasks, including semantic segmentation and object classification. Point convolutions are widely used to efficiently process 3D data representations such as point clouds or graphs. However, we observe that the receptive field size of recent point convolutional networks is inherently limited. Our dilated point convolutions alleviate this issue, they significantly increase the receptive field size of point convolutions. Importantly, our dilation mechanism can easily be integrated into most existing point convolutional networks. To evaluate the resulting network architectures, we visualize the receptive field and report competitive scores on popular point cloud benchmarks.**

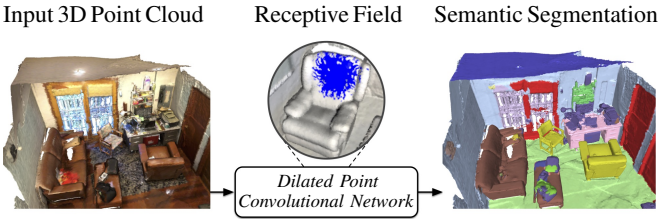

Fig. 1. This work presents *Dilated Point Convolutions* (DPC). We observe that existing point convolutional networks have inherently small receptive field sizes. Assisted by this observation, we compare different network architectures and propose our dilation mechanism as a simple yet elegant solution to significantly increase the receptive field size of point convolutions and improve their performance on multiple point cloud processing tasks.

I. Introduction

The past years have witnessed a tremendous development of 3D scene understanding methods on several tasks including semantic segmentation [18], object detection [32], and instance segmentation [5]. Recent advancements such as point convolutional layers [25], [26], [27] which can directly operate on 3D point clouds further boosted the field.

In the 2D image domain, analyzing the receptive field is an important tool for diagnosing and comprehending convolutional neural networks (CNN). The receptive field of a neural unit describes the region of the input data that influences its output value. All input data outside of the receptive field does not contribute to the output. Hence, large receptive fields are important since they enable reasoning on a larger input context.

Current successful architectures operating on grid-like data (*e.g.* images [22], [23], [8]), increase the receptive field implicitly by using *deeper* network architectures. However, only few works explicitly study the influence of receptive fields in the domain of 2D image CNNs [15], [17]. So far, there is no work analyzing the receptive fields of deep networks operating directly on 3D point clouds. Such a study is particularly challenging, since the theoretical size of receptive fields is difficult to compute due to the non-uniform structure of 3D point clouds. Nevertheless, we argue that the concept of receptive fields is equally important in the 3D domain.

Point convolutional layers [25], [26], [27] are a major driving force behind the success of networks that can directly operate on unstructured data such as 3D point clouds. Furthermore, they can be seen as a generalization of *discrete* convolutions. While continuous point convolutions operate on data sampled at continuous positions in space, discrete convolutions operate on grid-structured data such as images or voxel-grids, *i.e.* the data is sampled at discrete positions.

As such, we propose to visualize the receptive fields to analyze different network architectures and we present a thorough ablation study comparing several strategies which increase the receptive field of point convolutions. Specifically, we look at common strategies to increase the receptive field by 1) stacking convolutional layers and 2) using larger kernel sizes. By visually analyzing the extent of the resulting receptive fields, we notice that their influence still remains rather limited. Motivated by these observations, we propose *Dilated Point Convolutions* as a means to significantly increase the receptive field size of point convolutions.

The paper is structured as follows: We start by discussing current methods for 3D point cloud processing and existing works analyzing receptive fields on discrete convolutions. Then, we review *Point Convolutions* as an instance of continuous convolutions on 3D point clouds. Next, we describe and visualize well established methods for increasing receptive fields, which leads us to the derivation of *Dilated Point Convolutions*. Finally, in the experimental section, we compare the aforementioned strategies.

Our contributions are as follows: (1) We evaluate most commonly used strategies to increase the receptive fields in current methods using point convolutions. (2) We propose to visualize the receptive field of point convolutions to make educated network design choices. (3) From these observations, we derive *Dilated Point Convolutions* (DPC) as an elegant mechanism to significantly increase the receptive field size. (4) Using DPCs we are able to report competitive scores on the task of 3D semantic segmentation on S3DIS [1] and ScanNet [4] as well as shape classification on ModelNet40 [28].

1: All authors are with the Computer Vision Group, Visual Computing Institute at RWTH Aachen University in Aachen, Germany.

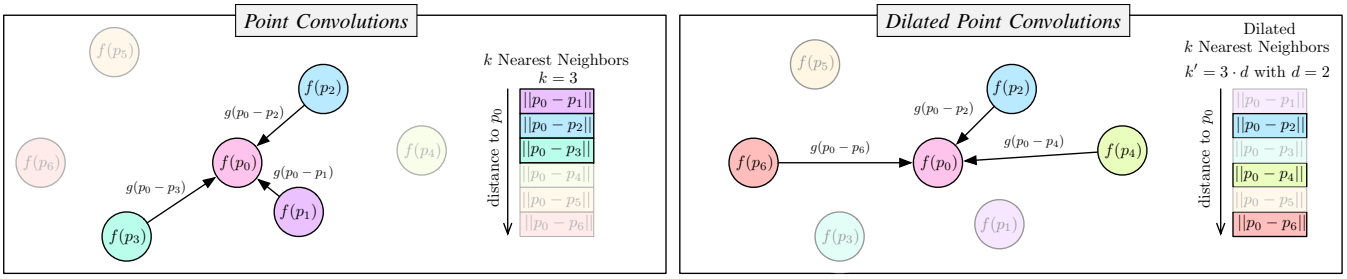

Fig. 2. *(Left)* **Point Convolutions.** Schematic illustration of point convolutions. The continuous feature function $f(\cdot)$ assigns a feature value to continuous point positions p. *(Right)* **Dilated Point Convolutions.** We propose *dilated* point convolutions as an elegant mechanism to significantly increase the receptive field of point convolutions resulting in a notable boost in performance at almost no additional computational cost (see Table IV). Instead of computing the kernel weights $g(\cdot)$ over the k nearest neighbors, we propose to compute the kernel weights over a *dilated* neighborhood obtained by computing the sorted $k \cdot d$ nearest neighbors and preserving only every d-th point.

II. Related Work

2D Projection Representation. Qi *et al.* [19] and Boulch *et al.* [2] project 3D point clouds into 2D representations, then apply 2D convolutional networks and finally fuse the results back into 3D space. These type of projections do not make use of the underlying geometric structure as they only operate on the projected appearance of the point clouds.

3D Volumetric Grid Representation. Maturana and Scherer [16] and Song *et al.* [28] voxelize point clouds into regular volumetric grids and apply 3D convolutions. These approaches are constrained by the fixed resolution of the 3D grid. Coarse grids lead to loss of detail and fine ones suffer from high memory and computational costs. The use of octrees [21] and kd-trees [10] offer improved grid resolutions. Recently, Graham *et al.* [7] offered a speed- and memory-efficient approach for sparse 3D convolutions which are applied only on occupied voxels. However, voxelized point clouds can still be problematic if adjacent points are far apart, which can hinder information flow.

3D Feature Learning on Point Sets. Numerous methods operate directly on 3D point clouds [18], [25], [26], [27]. They follow-up on the seminal work of *PointNet* [18] which applies point-wise multi-layer-perceptrons (MLP) followed by max-pooling over all points to extract a global point cloud descriptor but fails to capture local structure. Local structure is implicitly considered in 2D images and 3D voxels by using spatial grids. Filters that incorporate the information of the neighboring points in the grid are then learned. Numerous methods rely on similar types of spatial neighborhoods in an unstructured point cloud: Hua *et al.* [9] compute nearest neighbors on the fly and bin them into spatial cells before using fully convolutional networks. Landrieu and Boussaha [11] compute neighborhoods by over-segmenting 3D point clouds into superpoints. However, the most popular method used by [6], [14], [20], [26], [27] consists in computing the k nearest neighbors (KNN) of every point to represent its neighborhood. *EdgeConvs* [26] establish this neighborhood on the feature space while *PointConv* [27], *PointNet++* [20] and *PointCNN* [14] use the spatial coordinates. Engelmann *et al.* [6] use KNN in the feature space and k-means in the world coordinate system to create neighborhoods.

Receptive Field Analysis. Few works systematically study the influence of receptive fields on 2D image CNNs [15], [17]. In general, deeper networks which stack multiple layers of 2D convolutions have proven to work better [22], [23]. Dilated convolutions [30], previously introduced as *atrous* convolutions [3], used in 2D image semantic segmentation, allow to efficiently enlarge the receptive field of filters to incorporate larger context without increasing the number of model parameters. In this work, we propose a simple yet effective dilation mechanism for 3D point convolutions.

III. Approach

In this section, we formally define *point convolutions* and examine the importance of a large *receptive field size* in the context of 3D point cloud processing. We revisit existing strategies to increase the receptive field. Then, we propose our main contribution *dilated point convolutions*, an elegant yet easy technique to significantly increase the receptive field size of point convolutional networks.

A. Point Convolutions

Point convolutions can be formulated using the general definition of continuous convolutions in a D-dimensional space. Continuous convolutions are defined as

$$\big(f * g\big)(p_i) = \int_{-\infty}^{+\infty} f(p_j) \odot g(p_i - p_j)\ dp_j, \qquad (1)$$

where $\odot$ is the Hadamard-product of the continuous feature function $f : \mathbb{R}^D \rightarrow \mathbb{R}^F$ assigning a feature-vector $f(p_j) \in \mathbb{R}^F$ to each position $p_j \in \mathbb{R}^D$, and the continuous kernel function $g : \mathbb{R}^D \rightarrow \mathbb{R}^F$ mapping a relative position to a kernel weight. In the case of 3D point clouds, we have $D = 3$ and the feature vector could for example contain the point position, color, and normal such that $f(p) \in \mathbb{R}^9$, see Figure 2. In most practical applications, *e.g.* reconstructed 3D point clouds, the feature function f is not fully known since only a limited number N of point positions p_n are observed or even occupied. Using Monte-Carlo integration, the continuous convolution can then be approximated as

$$\big(f * g\big)(p_i) \approx \frac{1}{N} \sum_{n=1}^{N} f(p_n) \odot g(p_i - p_n), \qquad (2)$$

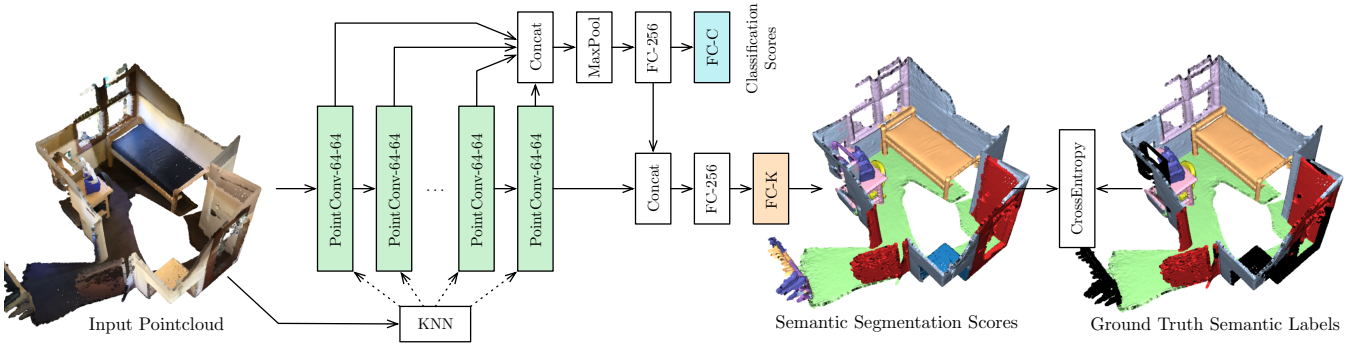

Fig. 3. Our model is built from a sequence of *point convolutional layers* (PointConv □, Section III-C). Fully connected layers are denoted by FC. The bottom output branch □ is used for the experiments on semantic segmentation. The top output branch □ is used for object classification. Each task is supervised using a cross-entropy loss, with either K semantic classes for semantic segmentation or C object classes for the object classification task.

where recent methods implement the kernel function $g(\cdot)$ as a learned parametric function based on a multi-layer perceptron (MLP)

$$g(p;\theta) = \text{MLP}(p\,;\theta), \tag{3}$$

where p is the relative position between two points and θ is a set of learned parameters. In order to extract high-frequency signals it is important to define localized kernels [31]. In 2D image CNNs, this is implemented by *e.g.* 3×3 or 5×5 pixel kernels. For point convolutions, this effect is achieved by limiting the cardinality of the local kernel support, *i.e.* by defining a local neighborhood $\mathcal{N}_i$ around each point p_i

$$\big(f * g\big)(p_i) \approx \frac{1}{|\mathcal{N}_i|} \sum_{p_k \in \mathcal{N}_i} f(p_k) \odot g(p_i - p_k). \tag{4}$$

The above definition of continuous convolutions is used in Wang *et al.* [25] and *PointConv* [27] which additionally proposes to weight the kernel function using the inverse local density to compensate for the non-uniform distribution of point samples. In *SpiderCNN*, Xu *et al.* [29] propose to replace the MLP by a combination of step functions and Taylor expansions to capture rich spatial information. A broader interpretation of continuous convolutions is used in *EdgeConv* [26], where the kernel function $g(\cdot)$ is not only defined over relative positions but also over the difference of learned point features. Independent of the concrete implementation, all previously mentioned methods, including *PointCNN* [14], rely on k nearest neighbors (KNN) to define a local neighborhood $\mathcal{N}$ resulting in local kernels. Next, after looking at the receptive field size, we use KNN neighborhoods to define *dilated* point convolutions.

B. Receptive Field Size.

A large receptive field is directly related to the performance of point convolutional networks (Section IV). Thus our goal is to increase the size of the receptive field. The receptive field (or *field of view*) of a neural unit within a deep network describes the region of the input point cloud that influences the output of that particular unit. In the context of 3D semantic segmentation, where the task is to assign a semantic label to each point in a given point cloud, the final decision on the label for a particular point is influenced only by those points which lie inside the receptive field of that particular point. All other points outside the receptive field do not contribute to the decision, see Figure 4. It is thus essential to design architectures with receptive fields large enough to cover the necessary context for each point.

A common approach to increase the receptive field size, similar to 2D architectures, consists in *stacking* multiple (point) convolutional layers. *EdgeConvs* [26] stack 3 convolutional layers, *SpiderCNN* [29] use 4 layers and *PCCN* [25] use 8. Here, we compare 3, 5 and 7 layers, see Table III.

Increasing the *kernel size* of the convolution is another popular technique. In the setup of point convolutions this effect is achieved by selecting a larger number k of nearest neighbors. Note, however, that this does not increase the number of model parameters since the kernel weights are computed over relative point positions using the parametric kernel function $g(\cdot)$, see Table III. This is in stark contrast to convolutions defined over discrete grid positions (*e.g.* 2D image CNN) where a larger kernel increases the number of model parameters.

C. Dilated Point Convolutions.

Using the previously mentioned approaches, the receptive field size still remains limited, see top 3 rows in Figure 4. Therefore, we propose *dilated point convolutions* (DPC) as an elegant yet efficient mechanism to increase the receptive field size. DPCs are equal to point convolutions (PC), however, they differ in the way they select neighboring points: While PCs directly use the k nearest neighbors, DPCs first compute the $k \cdot d$ nearest neighbors and then select every d-th neighbor, see Figure 2 (right). Note that for $d = 1$, DPCs are identical to PCs. The dilation causes a significantly increased receptive field size (see Figure 4). However, the number of parameters remains unchanged. The larger number $k \cdot d$ of neighbors that needs to be computed adds a sublinear computational overhead. See Table IV. Another positive aspect about DPC is that they can directly be added – with minimal modifications – to most existing point convolutional networks, if the local kernel neighborhood $\mathcal{N}$ originates from a nearest neighbor search.

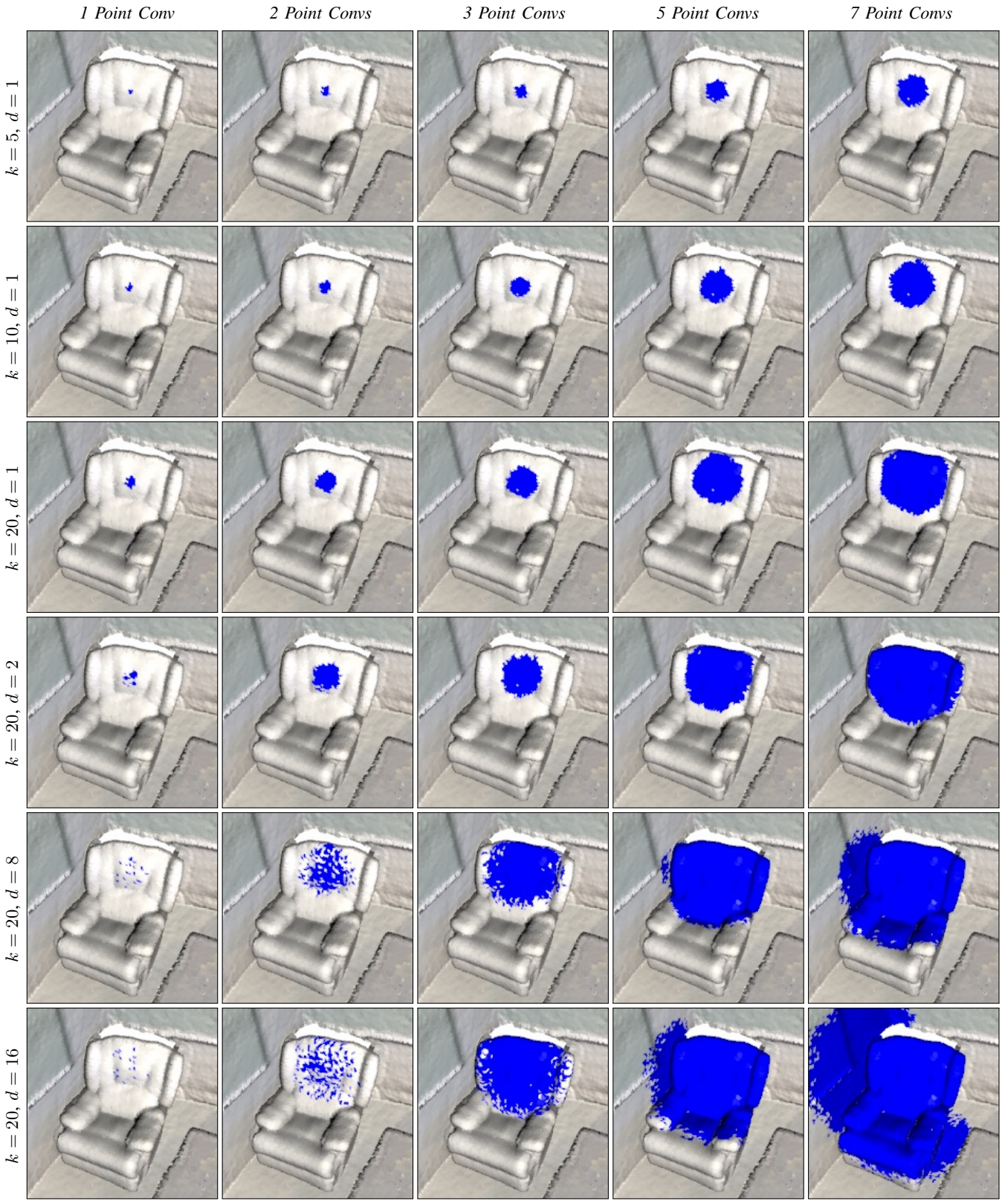

Fig. 4. **Receptive field** visualized in blue for different network architectures using an increasing number of *Point Convolutions* (columns) and increasing kernel sizes (rows) based on the number of nearest neighbors k and dilation factor d. The receptive field sizes of point convolutions without dilation ($d = 1$) are substantially smaller. However, for large dilations, *e.g.* $d = 16$ the receptive field is sparse in early stages of the deep network (bottom left).

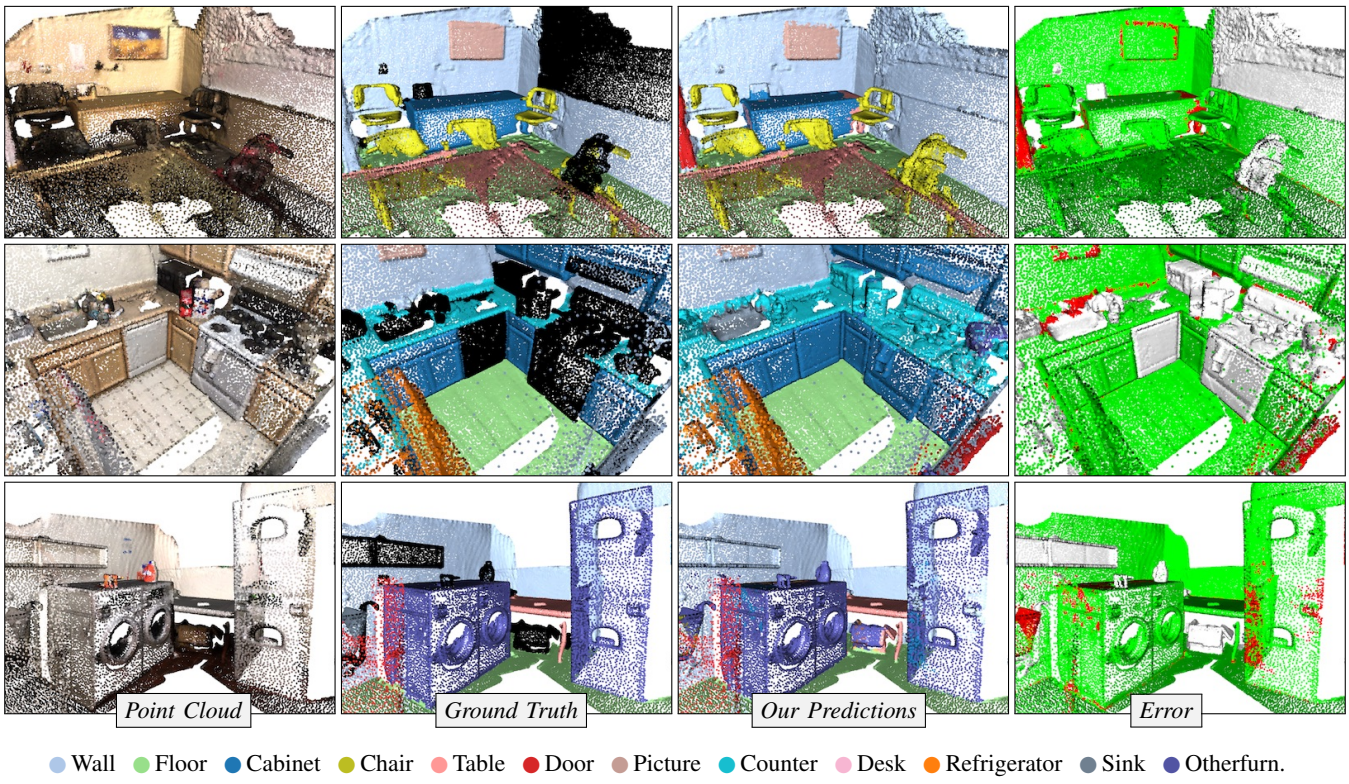

Fig. 5. **Results of our method on ScanNet v2 dataset [4] validation.** Left to right: Input RGB point cloud, semantic segmentation ground truth, semantic segmentation prediction and the error where green shows correct predictions, red shows wrong predictions and white indicates unlabeled ground truth.

IV. Experiments

Model Architecture. In all our experiments, we use a deep convolutional model as depicted in Figure 3. The main branch (shown in green) consists of stacked (dilated) point convolutions. The k nearest neighbors (KNN) for each point are computed on-the-fly. The final point features are concatenated with global features obtained by max-pooling over the concatenated point features at different depth-levels.

A. 3D Semantic Segmentation

Task and Metrics. The goal is to predict a semantic label for each point in a given point cloud. This task is especially well-suited to analyze the effectiveness of larger receptive fields, since the label of each point is only influenced by points in its receptive field. We adopt the commonly used metrics: mean intersection over union (mIoU), mean class accuracy (mAcc), and overall accuracy (oAcc).

Datasets. We evaluate on two datasets: (1) Stanford Large-Scale 3D Indoor Spaces (S3DIS) [1] contains dense 3D point clouds from 6 large-scale indoor areas, consisting of 271 rooms from 3 different buildings. The points are annotated with 13 semantic classes. We use the common train/test split, which trains on all areas except *Area 5* which we keep for testing [1], [24], [25]. (2) ScanNet v2 [4] contains 3D scans of a wide variety of indoor scenes, including apartments, hotels, conference rooms and offices. The dataset contains 20 valid semantic classes. We use the public training, validation and test split of 1201, 312 and 100 scans, respectively.

Training Details. We train our networks using the Adam optimizer and exponential-decay learning-rate scheduling. During training we randomly sample 4092 points from crops of 3 m side length. This differs from most concurrent methods which train on 1 m or 1.5 m crops. Since our model has a much larger receptive field it can learn to make use of this additional context. In general, small training crops could hinder the network to learn from larger context as soon as the size of the receptive field exceeds the size of the training crops. Points are sampled without replacement and we use zero-padding if there are less than 4092 points.

Results and Discussion We report scores of our best performing models on the ScanNet v2 dataset [4] and the S3DIS dataset [1] in Table I. Our dilated point convolutional model is able to outperform other recent KNN-based point convolutional networks by a significant margin on S3DIS,

TABLE I

3D Semantic segmentation on S3DIS (A5) and ScanNet V2.

	Method	mIoU	mAcc	oAcc
S3DIS Area 5	PointNet [18]	41.1	49.0	-
	KWYND [6]	52.2	59.1	84.2
	PointCNN [14]	57.3	63.9	85.9
	SPG [12]	58.0	66.5	86.4
	PCNN [25]	58.3	67.0	-
	DPC (Ours)	**61.28**	**68.38**	**86.78**
ScanNet	DPC (Val-set)	59.52	67.21	85.95
	DPC (Test-set)	59.2	-	-

TABLE II

OBJECT CLASSIFICATION SCORES ON MODELNET40

Method	# Points	oAcc	mAcc
PointNet[18]	1k	89.2	86.2
PointNet++(with normals)[20]	5k	91.9	-
Kd-Net[10]	32k	91.8	88.5
EdgeConv[26]	1k	92.2	90.2
SO-Net(with normals)[13]	5k	92.4	90.8
SpiderCNN(with normals)[29]	1k	92.4	-
DPC (Ours) with normals	4k	**93.1**	**91.4**

and provides competitive scores on ScanNet, specifically among point convolutional approaches. In Figure 5, we show qualitative results on the ScanNet validation dataset. We highlight wrong predictions in red (see right-most column).

B. 3D Object Classification

Dataset. ModelNet40 consists of CAD models that belong to one of 40 different categories. We use the official split of 9843 shapes for training and 2468 for testing, as in [20]. We randomly sample 4,000 points from the 3D model of an object. The input features are the 3D coordinates and the surface normals (6 input channels in total). *Comparison.* Table II shows the comparison between our method and prior methods. We report overall classification accuracy (oAcc) and mean classification accuracy (mAcc). Next, we present an ablation study of all model hyper-parameters.

C. Ablation Study

We perform an ablation study on the previously introduced mechanisms for increasing the receptive field size. The hyper-parameters that we analyze in particular are the number of point convolutional layers, the nearest neighbors k and the dilation factor d. The main results are presented in Table III and Table IV. The ablation studies are performed on Area 5 of the S3DIS dataset [1]. In the following, we discuss the influence of the individual parameters.

Depth and Number of Neighbors k (Table III). By increasing the number of convolutional layers, we can build deeper networks. Similar to discrete convolutions, deep point convolutional networks perform better than shallow ones.

TABLE III

ABLATION STUDY: STACKING POINT CONVOLUTIONS AND VARYING KERNEL SIZE k. DATASET: S3DIS AREA 5.

Number of PointConvs	Number of Neighbors k	Time per Forward-Pass	Number of Parameters	mIoU	mAcc
3	5	12.10 ms	$402 \cdot 10^3$	50.04	57.42
3	10	13.64 ms	$402 \cdot 10^3$	50.98	58.16
3	20	17.65 ms	$402 \cdot 10^3$	**52.25**	**60.83**
5	5	14.53 ms	$625 \cdot 10^3$	52.69	58.87
5	10	17.12 ms	$625 \cdot 10^3$	52.91	59.57
5	20	23.35 ms	$625 \cdot 10^3$	**53.27**	**60.15**
7	5	16.99 ms	$880 \cdot 10^3$	52.93	59.87
7	10	20.68 ms	$880 \cdot 10^3$	53.57	60.92
7	20	29.38 ms	$880 \cdot 10^3$	**53.93**	**61.73**

TABLE IV

ABLATION STUDY: DILATED POINT CONVOLUTIONS. VARYING DILATION FACTORS d. DATASET: S3DIS AREA 5.

Number of PointConvs	Number of Neighbors k	Time per Forward-Pass	Number of Parameters	Dilation d	mIoU	mAcc
7	20	29.38 ms	$880 \cdot 10^3$	1	53.93	61.73
7	20	31.57 ms	$880 \cdot 10^3$	2	55.83	61.76
7	20	35.36 ms	$880 \cdot 10^3$	8	**61.28**	**68.38**
7	20	51.65 ms	$880 \cdot 10^3$	16	58.79	65.84

Equally, the performance increases with the number of neighbors. However, increasing the number of neighbors increases the computational cost, resulting in slower inference times. Furthermore, increasing the number of convolutions leads to additional memory consumption.

Dilation Factor d (Table IV). Dilated Point Convolutions are an efficient tool to rapidly increase the receptive field of convolutions. Using dilation, the receptive field can be increased significantly (Figure 4) at constant memory requirements and a marginal increment in processing time. The improved performance on the semantic segmentation task shows that indeed a larger receptive field is important. However, the rapidly increasing receptive field resulting in large receptive fields in later layers is also responsible for sparsely sampled neighborhoods in earlier layers. We assume that this makes it harder for the network to learn high-frequency or local features. In future work, it could be interesting to investigate deep convolutional networks using Dilated Point Convolutions with a dilation rate d that increases with the depth of the network. Intuitively, such a network could learn localized signals in the earlier stages and higher level information at later stages.

Model Size. Note that, since the kernel function $g(p)$ is defined over relative point positions p, the number of trainable parameters is independent of the number of neighbors k (and hence the dilation factor d). As such, increasing the number of neighbors k (or the dilation factor d) increases the receptive field without increasing the model size.

V. CONCLUSION

In this work, we reviewed several mechanisms to increase the receptive field size of 3D point convolutions. We analyzed and compared different network architectures based on the receptive field size which we showed to be directly related to the performance of point convolutional networks. Specifically, we have proposed *dilated point convolutions* as an elegant and efficient technique to significantly increase the receptive field size of point convolutions. As a result, we were able to report solid improvements over well-known baseline methods for 3D semantic segmentation and 3D object classification. More importantly, our dilation mechanism can easily be integrated into most existing point convolutional networks. We hope these insights enable the research community to develop better performing models.

Acknowledgements: This work was supported by the ERC Consolidator Grant DeeViSe(ERC-2017-COG-773161). We thank Mats Steinweg, Dan Jia, Jonas Schult and Alexander Hermans for their valuable feedback.

REFERENCES

[1] Iro Armeni, Ozan Sener, Amir R. Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3D Semantic Parsing of Large-Scale Indoor Spaces. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] Alexandre Boulch, Joris Guerry, Bertrand Le Saux, and Nicolas Audebert. SnapNet: 3D point cloud semantic labeling with 2D deep segmentation networks. In *Computers & Graphics*, 2017.

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. In *International Conference on Learning Representations (ICLR)*, 2015.

[4] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[5] Cathrin Elich, Francis Engelmann, Jonas Schult, Theodora Kontogianni, and Bastian Leibe. 3D-BEVIS: Birds-Eye-View Instance Segmentation. In *German Conference on Pattern Recognition (GCPR)*, 2019.

[6] Francis Engelmann, Theodora Kontogianni, Jonas Schult, and Bastian Leibe. Know What Your Neighbors Do: 3D Semantic Segmentation of Point Clouds. In *European Conference on Computer Vision Workshop (ECCV'W)*, 2018.

[7] Benjamin Graham, Martin Engelcke, and Laurens van der Maaten. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise Convolutional Neural Network. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[10] Roman Klokov and Victor Lempitsky. Escape from Cells: Deep Kd-Networks for the Recognition of 3D Point Cloud Models. In *International Conference on Computer Vision (ICCV)*, 2017.

[11] Loïc Landrieu and Mohamed Boussaha. Point Cloud Oversegmentation with Graph-Structured Deep Metric Learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[12] Loïc Landrieu and Martin Simonovsky. Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[13] Jiaxin Li, Ben M Chen, and Gim Hee Lee. SO-Net: Self-Organizing Network for Point Cloud Analysis. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[14] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. PointCNN: Convolution On X-Transformed Points. In *Neural Information Processing Systems (NIPS)*, 2018.

[15] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the effective receptive field in deep convolutional neural networks. In *Neural Information Processing Systems (NIPS)*, 2016.

[16] Daniel Maturana and Sebastian Scherer. VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In *International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[17] Dmytro Mishkin, Nikolay Sergievskiy, and Jiri Matas. Systematic Evaluation of Convolution Neural Network Advances on the ImageNet. *Computer Vision and Image Understanding (CVIU)*, 2017.

[18] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[19] Charles R. Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and Multi-View CNNs for Object Classification on 3D Data. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[20] Charles R. Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Neural Information Processing Systems (NIPS)*, 2017.

[21] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. OctNet: Learning Deep 3D Representations at High Resolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[22] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations (ICLR)*, 2015.

[23] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going Deeper with Convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[24] Lyne P. Tchapmi, Christopher B. Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese. SEGCloud: Semantic Segmentation of 3D Point Clouds. In *International Conference on 3D Vision (3DV)*, 2017.

[25] S. Wang, S. Suo, W.C. Ma, A. Pokrovsky, and R. Urtasun. Deep Parametric Continuous Convolutional Neural Networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[26] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic Graph CNN for Learning on Point Clouds. In *ACM Transactions on Graphics (TOG)*, 2018.

[27] Wenxuan Wu, Zhongang Qi, and Fuxin Li. PointConv: Deep Convolutional Networks on 3D Point Clouds. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[28] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D ShapeNets: A Deep Representation for Volumetric Shapes. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[29] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao. SpiderCNN: Deep Learning on Point Sets with Parameterized Convolutional Filters. In *European Conference on Computer Vision (ECCV)*, 2018.

[30] Fisher Yu and Vladlen Koltun. Multi-Scale Context Aggregation by Dilated Convolutions. In *International Conference on Learning Representations (ICLR)*, 2016.

[31] Matthew D. Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *European Conference on Computer Vision (ECCV)*, 2014.

[32] Yin Zhou and Oncel Tuzel. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.