Vicinity-based Abstraction: VA-DGCNN Architecture for Noisy 3D Indoor Object Classification

Jakub Walczak[1][0000−0002−5632−9484], Adam Wojciechowski[1][0000−0003−3786−7225], Patryk Najgebauer[2][0000−0002−7168−3019], and Rafał Scherer[2][0000−0001−9592−262X]

[1] Institute of Information Technology, Lodz University of Technology, Łódź, Poland
{jakub.walczak@p.lodz.pl, adam.wojciechowski}@p.lodz.pl
[2] Department of Intelligent Computer Systems, Częstochowa University of Technology
Al. Armii Krajowej 36, 42-200 Częstochowa, Poland
{patryk.najgebauer, rafal.scherer}@pcz.pl

Abstract. One of the outstanding benchmark architectures for point cloud processing with graph-based structures is Dynamic Graph Convolutional Neural Network (DGCNN). Though it works well for classification of nearly perfectly described digital models, it leaves much to be desired for real-life cases burdened with noise and 3D scanning shadows. Therefore we propose a novel, feature-preserving vicinity abstraction (VA) layer for the EdgeConv module. This allowed for enriching the global feature vector with the local context provided by the k-NN graph. Rather than processing a point together with its neighbours at once, local information is aggregated before further processing, unlike in the original DGCNN. Such an approach enabled a model to learn accumulated information instead of max-pooling features from local context at the end of each EdgeConv module. Thanks to this strategy mean- and overall classification accuracy increased by 9.4pp and 4.4pp, respectively. Furthermore, thanks to processing aggregated information rather than the entire vicinity, the new VA-DGCNN model converges significantly faster than the original DGCNN.

Keywords: 3D object classification, point clouds, DGCNN

1 Introduction

Classification of real three-dimensional objects, registered as a point cloud, is currently a hot research topic, mainly due to the popularity of ubiquitous depth sensors and emerging applications of 3D data automatic processing. Whereas artificial, synthetic point sets [14] are handled quite well by the current benchmark methods, processing real-world data, collected in non-laboratory poorly constrained environments, still leaves much to be desired. Such data imposes

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

many challenges [12], usually not related to synthetic data such as uneven object sampling, noise presence, missing points resulting from scanning shadows, or remarkable classes imbalance, are just cases in point.

A challenge with approximating a function of point clouds is twofold: processing imperfect real-life data and achieving immutable results regardless of any permutation of point ordering. In other words, 3D object spatial point distribution can be depicted by any permutation of data. Unlike in the case of images, sound waves, or other well-ordered structures, point clouds imply permutation equivariance property of a model. As pointed out by the authors of [11], representation of a function on sets is burdened with some limitations out of which the essential one is the minimum size of latent space required to make a model able to represent and generalize a set. The authors concluded that the latent space dimension should be at least as large as the input size of a set [11].

Due to substantial cardinality of point cloud data sets, their efficient classification is recently performed with graph-based approaches. Such a hierarchical point feature regression allows for retrieving core features constellation and performing efficient classification afterwards. Nevertheless, incomplete and noised real-world data sets impose severe challenges on a core features aggregation process. In the paper, we aim at improving the performance of real-life 3D data classification. To this end, we developed a novel method for processing real-life three-dimensional objects described by point clouds. The proposed method introduces multi-layer perceptron-based (MLP-based) features integration rather than a coarse statistical features aggregation. Such a subtle modification exceeds the accuracy of the state-of-the-art methods on an established real-world data set (S3DIS [1]).

Compared to the literature, our method is faster as it uses vicinity-aware aggregation of the local context. At the same time, it demonstrates better accuracy because it learns more descriptive features. These studies provide new insights, showing that the noisy, incomplete point clouds require aggregation of the local context before hierarchical features learning. We also investigate the associated problem of a local context range determination and show the tradeoff between the size of the neighbourhood and the processing speed.

The rest of the paper is organized as follows. Section 2 presents related works and the DGCNN architecture. A novel VA-DGCNN method is proposed in Section 3. Research methodology, in turn, is outlined in Section 4. Section 5 contains results of experiments. The article is concluded in Section 6.

2 Related Works

Point clouds are one of the simplest representation of 3D shapes coming, for example, from LIDARs or time-of-flight cameras. Current methods allow processing point clouds directly without intermediate mesh or denoising. Neural networks innate input are regular structures (sequences, vectors, images or volumes), whereas point clouds are unstructured and of various size. The first network able to process such data was PoinNet [7]. PointNet takes a point from a

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

point cloud and uses max pooling symmetric function. Qi et al. [8] in their further study elaborated an extension of the basic PointNet network, introducing hierarchical neural network for processing local features with a new approach called PointNet++.

Since then, among point cloud classification-driven deep neural network models, graph-based solutions became one of the most efficient ones. As they surpassed the previous approaches, we compare our method in Section 4 only with these state-of-the-art graph-based networks. Graph-based networks usually consider points as vertices of a graph and construct then directed graph edges relying on features of a point vicinity. More than often, such methods perform convolution and pooling operation in spatial domain. Convolution is normally implemented within MLP-inspired module over spatial neighbourhood, whereas pooling aggregates information from a vicinity by forming a new coarsened graph. A graph spread over a point cloud reflects well its sparsity and characteristics. Moreover, it is inherently permutation-invariant, assuring nontrivial property crucial for unorganised point cloud classification [9]. In consequence, dominant graph-based approaches [4] like DGCNN [13] or closely related LDGCNN [15] exceed 92% for overall accuracy and 90% for mean accuracy on the well-known benchmark data set ModelNet40 [14]. Other competitive graph-based models like HGCNN [16], Dynamic Points Agglomeration Module (DPAM) [5], Kernel Correlation Network (KCNet) [10], or ClusterNet [2] – utilizing rigorously rotation-invariant module to extract point rotation-invariant features, can also be analysed; however they reveal inferior classification accuracy in comparison with DGCNN [4].

The classification accuracy deteriorates when performing on non-synthetic, real world data sets like S3DIS [1]. Such a database of real scanned objects, affected by uneven object sampling, noise presence, missing points – resulting from scanning shadows, imposes severe disturbances reducing quality of semantic segmentation $mIoU$ to about 56% or 64% for DGCNN and DPAM respectively [4].

Intrigued by real data (S3DIS) experiments of semantic segmentation, we conducted a study on the classification accuracy of such data. It revealed that for S3DIS database, DGCNN substantially deteriorates classification accuracy, achieving barely 60.2% and 80.3% for mean accuracy and overall accuracy respectively. It confirmed our assumptions about the vulnerability of DGCNN to noised and incomplete data. We claim that the drastic deterioration of classification accuracy results from coarse, averaged aggregation of neighbouring points features. For supporting the above hypothesis, we selected the DGCNN network as it is a well-established baseline solution with still superior accuracy.

DGCNN is an architecture constituted by stacked layers of, so-called, EdgeConv modules. They are, in turn, followed by MLP and MaxPool layers. A single EdgeConv module selects at first k nearest neighbours for each analysed point. Then, vectors between point's neighbours and a point itself, as well as points' centroid coordinates itself, are processed by MLP (see Fig. 1).

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

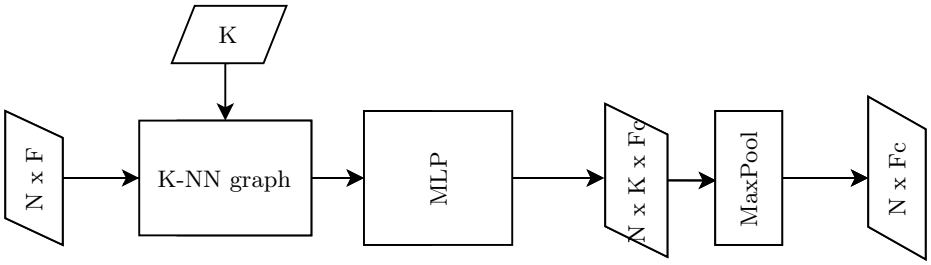


Fig. 1: EdgeConv layer of DGCNN architecture. The input is a matrix of N points of F features (coordinates), k denotes the number of nearest neighbours used to construct the graph, and F_c is the number of resulting features on the layer output.

A k-NN graph for the input point cloud $\mathcal{D}$ of N points and F features/coordinates, is defined as a matrix $\mathbf{V}$ of size $N \times k \times F$. For the first iteration, when $F = 3$ (features are input coordinates) an entry for $j-$th neighbour of $i-$th point is denoted as: $\mathbf{V}_{ij} = \{\mathbf{x}_j^{(x)}, \mathbf{x}_j^{(y)}, \mathbf{x}_j^{(z)}\}$ such that $||\mathbf{x}_i - \mathbf{x}_j||_2 \leq ||\mathbf{x}_i - \mathbf{V}_{i(j+1)}||_2$. MLP is defined as in Equation 1.

$$mlp(\cdot) = ReLU(bn(\cdot * G)) \tag{1}$$

where G is a matrix of convolution kernels, $bn(\ \cdot\ )$ is a batch normalization, namely, scaling throughout batches, and $ReLU$ is defined as $ReLU(\cdot) = \max\{0, \cdot\}$

Later on, the maximum feature out of all neighbours is extracted for each point. Out of resulting values, a vector of length F_c of maxima throughout neighbours is drawn for each analysed point. Such an approach lets a model learn local context well. However, it takes local information into consideration in the last but one layer, MaxPool, where the vicinity context is aggregated into a single representative per point. We argue that processing a point and the associated vicinity information at once, may deteriorate classification results due to the loss accumulated information provided by the local context. Introducing an additional layer of abstraction may enrich global feature vector with local information at a scale [3] – here defined by the vicinity of size k. Additionally, such vicinity-aware feature aggregation may speed up the model training.

3 Proposed Method

The classical DGCNN is constructed by stacked layers of edge-convolution modules (EdgeConv, see Fig. 1), followed by a multilayer perceptron, where the maximum value of features throughout the entire vicinity k is extracted for the every point $\{p_1, p_2, p_3, ..., p_N\}$. In such an architecture, features are processed without any regard to local context provided by the vicinity until the last but

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

one layer, i.e. MaxPool, which, in the simplest manner, aggregates local context to a single value per point.

In order to take into consideration the vicinity context from the very beginning, we decided to introduce a simple single-layer module of shared vicinity abstraction (VA) (Eq. 2) just between the k-NN graph and MLP (see Fig. 2). VA acts as if it combines neighbours' features so that the information is aggregated in place and more general traits might be retrieved for further processing. As such, it may be thought of as generating a kind of overlapping super-points for each patch defined by a point and its vicinity

$$\mathcal{D}_{VA} = VA(\mathbf{V}) = \mathrm{ReLU}(\mathbf{W}^T\mathbf{V} + \mathbf{h}) \,, \tag{2}$$

where $\mathcal{D}_{VA}$ is a point cloud of points after VA layer, $\mathbf{W}$ are weights associated with the layer, $\mathbf{V}$ is a matrix of neighbours of dimensions ($N \times k \times F$), and $\mathbf{h}$ is the layer bias.

$$\mathcal{D}' = \mathrm{EdgeConv}(\mathcal{D}, k) \,, \tag{3}$$

where k is an assumed number of nearest neighbours. After VA layer, each point

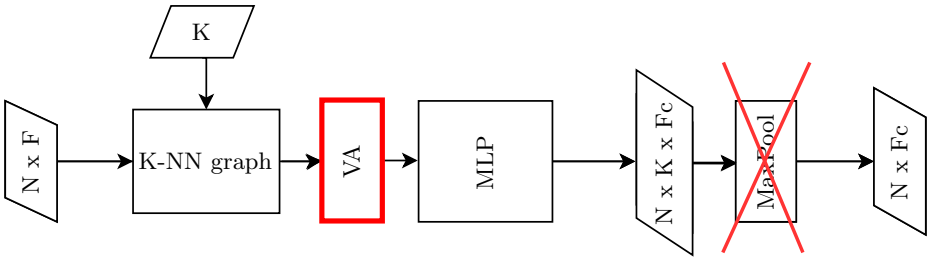


Fig. 2: Proposed modification of EdgeConv layer taking into account the vicinity context before features processing.

of the input point cloud $\mathcal{D}$ is transformed into multidimensional feature space: $\mathcal{D}_{VA} = \{\mathbf{x}_1^{VA}, \mathbf{x}_2^{VA}, \mathbf{x}_3^{VA}, ..., \mathbf{x}_N^{VA}\}$ where $\mathbf{x}_i^{VA} \in \mathbb{R}^F$ is a resulting vector of features (Eq. 2, Fig. 3) which is passed to convolution layers. This describes a single EdgeConv module where an input point cloud matrix $\mathcal{D}$ of size $N \times F$ (N points of F initial features) is transformed to vicinity-aware point cloud matrix $\mathcal{D}'$ of size $N \times F_c$ (Eq. 3). Besides classification quality boosting, such a strategy will allow a model to converge faster as instead of processing the entire vicinity, we process only combined features extracted from that vicinity. Such features carry more semantic information with lower memory requirements.

It should be noted that VA is a highly permutation-changeable module. There is, however, stipulated an intrinsic order relation among points by means of a distance metric used to collect nearest neighbours (Euclidean in this context). Therefore, VA does not need to show permutation-equivariant properties as neighbours of a point do not permute for that point.

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

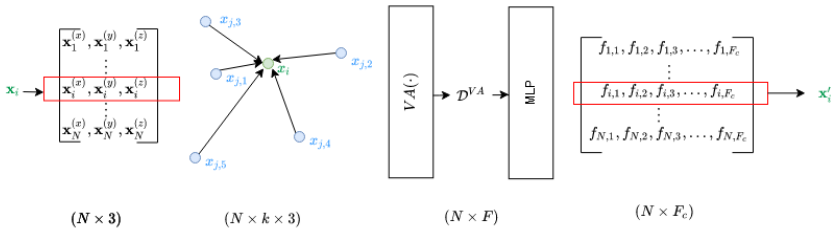


Fig. 3: Shapes of matrices (bottom row) while processing through *EdgeConv* with VA

4 Research Methodology

In this section we compare our approach with DGCNN on the real-life S3DIS dataset [1]. The rationale behind comparing only with DGCNN is that it is superior to the PointNet NN family and every other previous method on every 3D data set.

4.1 Used data

Classification models often are evaluated on either ModelNet10 or ModelNet40 [14] data sets. In these popular benchmark data sets, individual objects instances (point clouds), were constructed by means of sampling digital, CAD models. These data sets seem to be not useful for evaluation of real-life application models due to the fact they consist of synthetic, noiseless, complete object structures rather than depth data acquisition samples, affected by numerous disturbances.

Therefore, we decided to use S3DIS [1] database collected with the Matterport Camera on six different areas (locations) having, in total, 273 single room scans labeled with respect to instances of 14 classes (see Fig. 4). This set represents real data raising real problems, like noise presence or scanning shadows, just to name a few. Moreover, S3DIS is a database of extremely uneven distribution of labels (see Fig. 5). More than 39% of data is tagged as *clutter*, i.e. an indefinite object type not belonging to any of 13 remaining classes. On the other hand, class *stairs* has barely 0.2% of representatives. Some classes have distinguishable geometrical context. On the other hand, three classes: *door*, *window*, *wall* are significantly less geometrically distinguishable than others, thus more challenging for classification.

For regularization purposes dataset was augmented by rotating points along vertical axis and by shifting.

4.2 Evaluation metrics

In order to indicate competitive advantages reached by the proposed method, standard evaluation metrics for point cloud classification were used [4]. Two

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

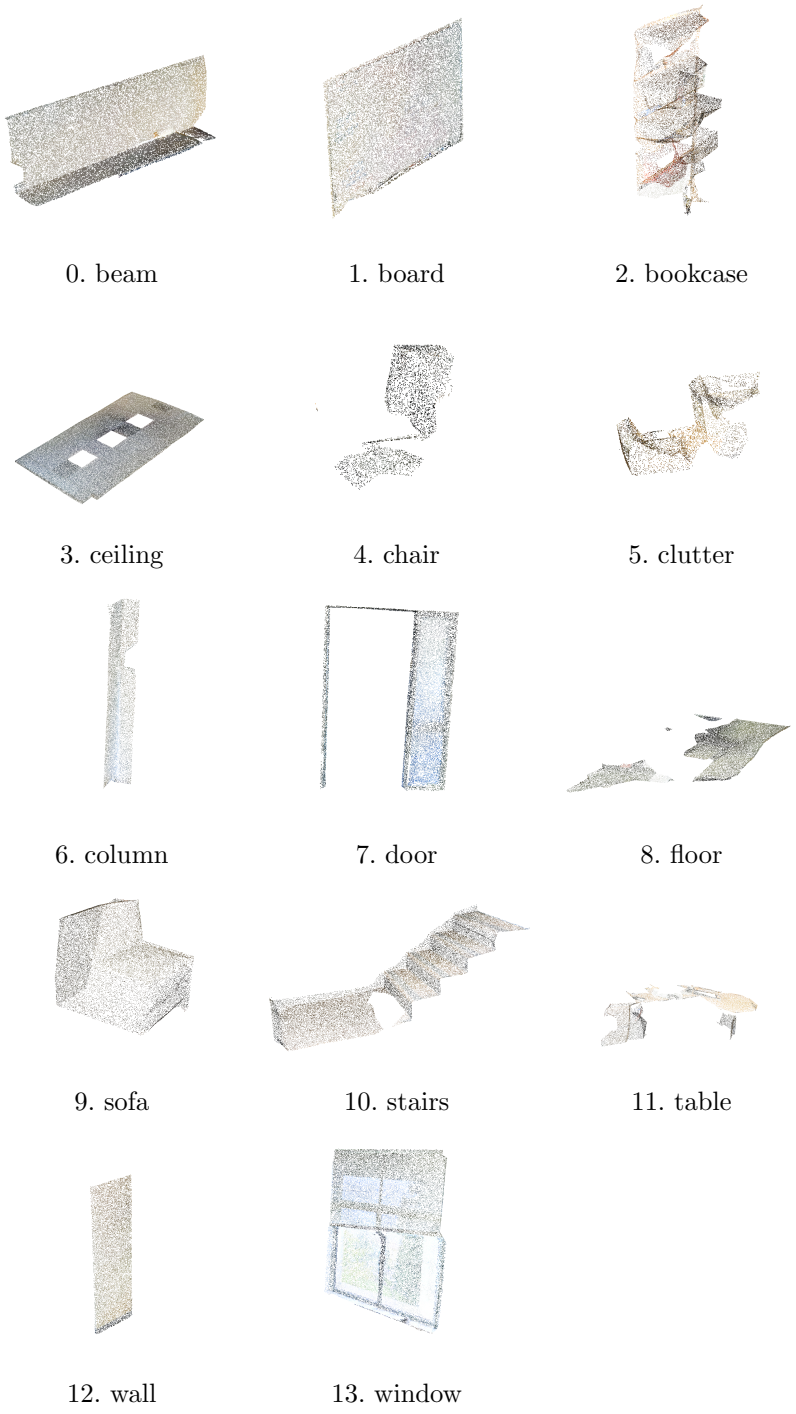


Fig. 4: Examples of class instances present in the S3DIS data set with their reference numbers (0 - 13).

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

Fig. 5: Histogram of classes across the S3DIS database showing its unbalance.

measures, namely mean accuracy (mAcc) and overall accuracy (oAcc) can be easily derived from class-wise confusion matrix $\mathcal{C}$ of size $A \times A$ (see Eq. 4 and 5).

$$mAcc = \sum_{i \leq A} \frac{\mathcal{C}_{i,i}}{\sum_{j \leq A} \mathcal{C}_{i,j}} \ , \tag{4}$$

where i is an iterator over rows (the actual labels) of a confusion matrix $\mathcal{C}$ and j is an iterator over columns (the predicted labels).

$$oAcc = \frac{\sum \mathrm{diag}(\mathcal{C})}{\sum_{i,j} \mathcal{C}_{i,j}} \ , \tag{5}$$

where $0 \leq i, j \leq A$. It should be noted that for data sets of extremely non-uniform classes distribution, $mAcc$ is more informative in drawing general conclusions. Such a situation occurs in the S3DIS database, where classes are relatively unbalanced. One of the classes (*clutter*) is on average about ten times over-represented than others. For multi-class classification mean- and overall accuracies may be thought to represent the same aspects of classification as macro- and micro-averaged F1-score respectively, which means that false positives and false negatives are taken into account as well.

For training deep learning models not only accuracy itself is crucial for evaluation but also the time required by a model to converge. This is the reason why in Tab. 2 the convergence time (T_c) is also juxtaposed throughout the methods being compared.

4.3 Experiments

Experiments were conducted using 5-fold cross validation. Prior to model training, the entire S3DIS database was split into three subsets: a training one, a

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

validation one, and a test one, respectively of 60%, 20%, and 20%. Splits were generated randomly for each fold, yet keeping the original distribution of classes (see Fig. 5) in each split. Splits were created at the beginning and each object was sampled to contain $1,024$ points. Sampling was done by random drawing with replacement. The same data set was used for the state-of-the-art DGCNN and the proposed novel VA-DGCNN model. During the training phase, each object was randomly permuted every time prior to passing it to the network. All hyper-parameters and the general complexity were the very same for both DGCNN and VA-DGCNN (Tab. 1). The solution was implemented in Py-

Table 1: Hyperparameters used for both DGCNN and VA-DGCNN. We perfomed experiments for various k and chose $k = 10$ as the best accuracy-speed tradeoff

Hyperparameter	**Value**
learning rate	0.0005
batch size	32
k	10
1. EdgeConv features	[3, 64, 64, 64]
2. EdgeConv features	[64, 128]
Classification module neurons	[128, 1024, 512, 256]

Torch [6]. Sufficiently small learning rate (see Tab. 1) was used to avoid gradient explosion. First EdgeConv layer uses convolutions of filters 3×1 for the first convolution and 64×1 for the rest. For the second EdgeConv filters of size 64×1 and 128×1 were applied. Classifier is built as four-layer MLP with 128, 1024, 512, 256 hidden neurons respectively. Learning was conducted for relatively long time to be sure it converged. Moreover, quality measures for the validation set were tracked to be sure that the model does not overfit.

Besides that studies of the impact of the vicinity size on the classification results were performed. Several possible vicinities k = 2, 5, 10, 15, 20, 25 were tested as to determine the optimal value with the most reasonable time to quality metrics tradeoff.

5 Results

As it might be noticed in Figure 6, vicinity increase causes nearly logarithmic increase of quality metrics at the cost of linear time growth. An inflection point is located for $k = 25$. For wider neighbourhood quality measures tend to lower. In fact, the neighbourhood of size $k = 10$ may be said to be the optimal one as it ensures high $mAcc$ and $allAcc$ keeping processing time in acceptable bounds. In

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

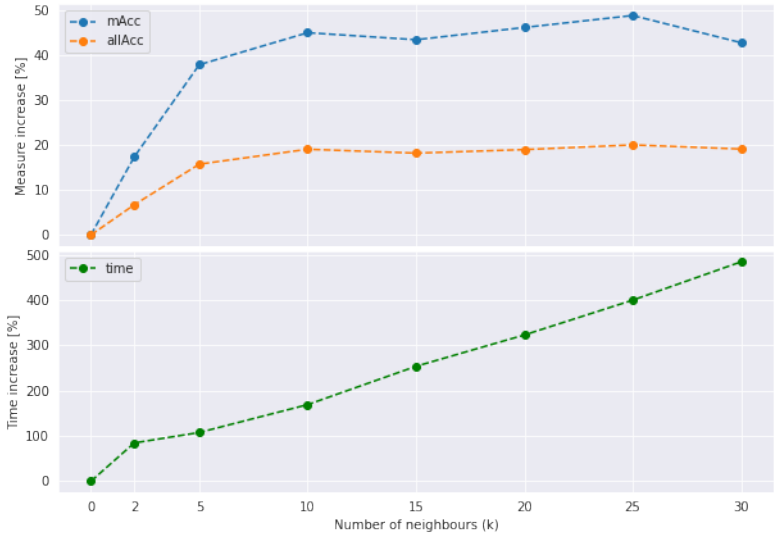


Fig. 6: *oAcc* and *mAcc* values for a test split vs. vicinity size (upper chart). Time vs. vicinity size (lower chart)

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

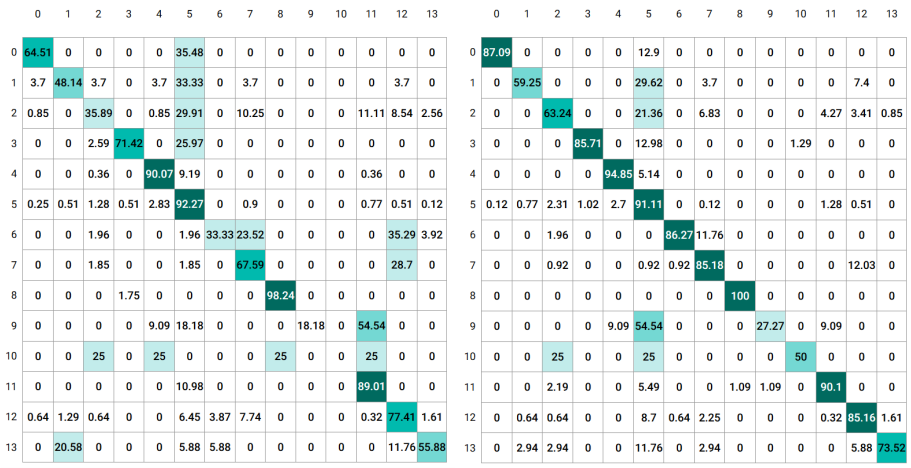
Table 2 a juxtaposition of evaluation measures for DGCNN and VA-DGCNN was presented. As it may be seen therein, the VA component enhanced remarkably

Table 2: Comparison between the benchmark DGCNN method and the novel VA-DGCNN with the proposed vicinity abstraction component

Measure		**DGCNN**	**VA-DGCNN**
$mAcc$	[%]	72.8 ± 7.7	82.2 ± 2.6
$oAcc$	[%]	85.7 ± 3.1	90.1 ± 0.9
T_c	[$\cdot 10^3$ sec]	9.7 ± 0.2	4.6 ± 0.2

the results of classification on the S3DIS data set. Mean accuracy $mAcc$, which is a more informative indicator for non-balanced data sets, yields the value of 82.2% which was an improvement by 9.4pp with respect to the original DGCNN. Also, overall accuracy ($oAcc$) improved by 4.4pp. This confirmed better classification results also for classes having much more representatives than the others (see confusion matrix in Fig. 7). Looking at the confusions matrices (Fig. 7), one may clearly see that confusing objects like *door*, *wall*, or *window* were better distinguishable having applied the proposed VA module. Also classes of a few samples, like *stairs* or *sofa* were better classified in the VA-DGCNN architecture. There was still a confusion between *board* and *bookcase*, yet diminished for the proposed VA-based solution. Having a look at the last row of Table 2 one may notice that the VA-DGCNN converged faster by virtually 53% (see also Fig. 8).

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19



	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	64.51	0	0	0	0	35.48	0	0	0	0	0	0	0	0
1	3.7	48.14	3.7	0	3.7	33.33	0	3.7	0	0	0	0	3.7	0
2	0.85	0	35.89	0	0.85	29.91	0	10.25	0	0	0	11.11	8.54	2.56
3	0	0	2.59	71.42	0	25.97	0	0	0	0	0	0	0	0
4	0	0	0.36	0	90.07	9.19	0	0	0	0	0	0.36	0	0
5	0.25	0.51	1.28	0.51	2.83	92.27	0	0.9	0	0	0	0.77	0.51	0.12
6	0	0	1.96	0	0	1.96	33.33	23.52	0	0	0	0	35.29	3.92
7	0	0	1.85	0	0	1.85	0	67.59	0	0	0	0	28.7	0
8	0	0	0	1.75	0	0	0	0	98.24	0	0	0	0	0
9	0	0	0	0	9.09	18.18	0	0	0	18.18	0	54.54	0	0
10	0	0	25	0	25	0	0	0	25	0	0	25	0	0
11	0	0	0	0	0	10.98	0	0	0	0	0	89.01	0	0
12	0.64	1.29	0.64	0	0	6.45	3.87	7.74	0	0	0	0.32	77.41	1.61
13	0	20.58	0	0	0	5.88	5.88	0	0	0	0	0	11.76	55.88

(a) DGCNN

	0	1	2	3	4	5	6	7	8	9	10	11	12	13
0	87.09	0	0	0	0	12.9	0	0	0	0	0	0	0	0
1	0	59.25	0	0	0	29.62	0	3.7	0	0	0	0	7.4	0
2	0	0	63.24	0	0	21.36	0	6.83	0	0	0	4.27	3.41	0.85
3	0	0	0	85.71	0	12.98	0	0	0	0	1.29	0	0	0
4	0	0	0	0	94.85	5.14	0	0	0	0	0	0	0	0
5	0.12	0.77	2.31	1.02	2.7	91.11	0	0.12	0	0	0	1.28	0.51	0
6	0	0	1.96	0	0	0	86.27	11.76	0	0	0	0	0	0
7	0	0	0.92	0	0	0.92	0.92	85.18	0	0	0	0	12.03	0
8	0	0	0	0	0	0	0	0	100	0	0	0	0	0
9	0	0	0	0	9.09	54.54	0	0	0	27.27	0	9.09	0	0
10	0	0	25	0	0	25	0	0	0	0	50	0	0	0
11	0	0	2.19	0	0	5.49	0	0	1.09	1.09	0	90.1	0	0
12	0	0.64	0.64	0	0	8.7	0.64	2.25	0	0	0	0.32	85.16	1.61
13	0	2.94	2.94	0	0	11.76	0	2.94	0	0	0	0	5.88	73.52

(b) VA-DGCNN

Fig. 7: Confusion matrices for the original and the proposed model. Rows represent the actual labels and columns predicted labels, respectively. The values of the confusion matrix are expressed as percent by row.

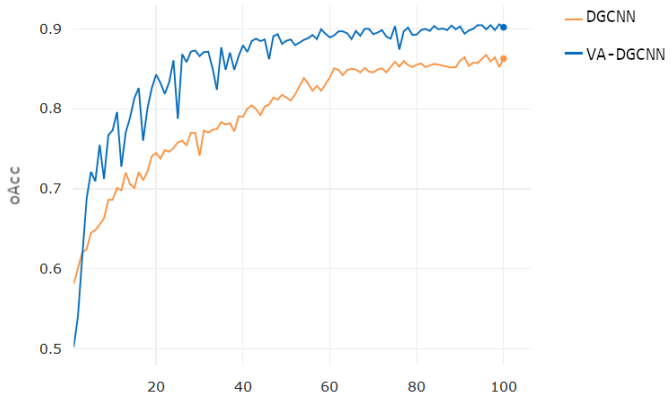


Fig. 8: *oAcc* values for validation split for a single fold

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

6 Conclusions

The results of the conducted experiments clearly confirmed what stated at the beginning of the paper, namely, aggregation of the local context prior to the actual features distribution learning improves results of object classification on noisy, real-life database like S3DIS. Thanks to the learning of aggregated local vicinity features, training process converges faster and more descriptive features are learnt. By applying the proposed, simple vicinity-abstraction layer, many benefits were reached. At first, thanks to the applied VA module, it was possible to boost *mAcc* by 9.4pp and *oAcc* by 4.4pp. Secondly, the convergence time decreased by more than a half, from around 9,700 sec to 4,600 sec. It is also clear that provided evidences relates only to indoor scans and outdoor point clouds would need further investigation due to dramatically different characteristic. As a recap, it was confirmed that enriching the global matrix with the aggregated features of the local context, enhanced the results of classification. An interesting aspect that will be considered in further studies relates to semantic segmentation of noisy data making use of the proposed strategy.

Acknowledgement

The research was co-funded by the Polish National Center for Research and Development under the LIDER XI program.

References

[1] Armeni, I., Sax, A., Zamir, A.R., Savarese, S.: Joint 2D-3D-Semantic Data for Indoor Scene Understanding. ArXiv e-prints (February 2017)
[2] Chen, C., Li, G., Xu, R., Chen, T., Wang, M., Lin, L.: Clusternet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 4994–5002
[3] Fang, H., Lafarge, F.: Pyramid scene parsing network in 3D: Improving semantic segmentation of point clouds with multi-scale contextual information. ISPRS Journal of Photogrammetry and Remote Sensing **154** (2019) 246–258
[4] Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M.: Deep learning for 3D point clouds: A survey. arXiv preprint arXiv:1912.12033 (2019)
[5] Liu, J., Ni, B., Li, C., Yang, J., Tian, Q.: Dynamic points agglomeration for hierarchical point sets learning. In: Proceedings of the IEEE International Conference on Computer Vision. (2019) 7546–7555
[6] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in PyTorch. (2017)
[7] Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2017) 652–660
[8] Qi, C.R., Yi, L., Su, H., Guibas, L.J.: PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In: Advances in neural information processing systems. (2017) 5099–5108

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19

[9] Segol, N., Lipman, Y.: On universal equivariant set networks. arXiv preprint arXiv:1910.02421 (2019)
[10] Shen, Y., Feng, C., Yang, Y., Tian, D.: Mining point cloud local structures by kernel correlation and graph pooling. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2018) 4548–4557
[11] Wagstaff, E., Fuchs, F.B., Engelcke, M., Posner, I., Osborne, M.: On the limitations of representing functions on sets. arXiv preprint arXiv:1901.09006 (2019)
[12] Walczak, J., Poreda, T., Wojciechowski, A.: Effective Planar Cluster Detection in Point Clouds Using Histogram-Driven Kd-Like Partition and Shifted Mahalanobis Distance Based Regression. Remote Sensing **11**(2465) (2019)
[13] Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph CNN for learning on point clouds. ACM Transactions on Graphics (tog) **38**(5) (2019) 1–12
[14] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3D ShapeNets: A deep representation for volumetric shapes. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (June 2015)
[15] Zhang, K., Hao, M., Wang, J., de Silva, C.W., Fu, C.: Linked dynamic graph CNN: Learning on point cloud via linking hierarchical features. arXiv preprint arXiv:1904.10014 (2019)
[16] Zhang, Y., Rabbat, M.: A graph-CNN for 3D point cloud classification. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE (2018) 6279–6283

ICCS Camera Ready Version 2021
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-77967-2_19