Evaluation of Input Sampling Methods for Deep-Learning-Based Semantic Segmentation of Large-Scale 3D Point Clouds

Jorge Francisco Ciprián-Sánchez[1], Josafat-Mattias Burmeister[2], Rico Richter[2], Gilberto Ochoa Ruiz[3], Jürgen Döllner[1]

[1] University of Potsdam, Hasso Plattner Institute, Germany - (jorge.cipriansanchez, juergen.doellner)@hpi.de
[2] University of Potsdam, Germany - (burmeister, rico.richter.1)@uni-potsdam.de
[3] Tecnológico de Monterrey, Mexico – gilberto.ochoa@tec.mx

Keywords: 3D Point Clouds, Deep Learning, Semantic Segmentation, Input Sampling

Abstract

3D point clouds used in geospatial applications typically contain billions of points. Processing 3D point clouds of this size as a whole with deep learning models requires computational resources (e.g., GPU memory) that are usually not available. To obtain 3D point clouds that can be processed by deep learning models, sampling methods that produce local subsets of large-scale 3D point clouds with a smaller extent or lower density are essential. Nonetheless, the impact of different input sampling methods on the semantic segmentation performance of deep learning models has received little attention so far. In this paper, we compare three widely used input sampling techniques (random sampling, farthest point sampling, and grid sampling) concerning the semantic segmentation performance of different deep learning architectures, using inputs of different spatial extents. We consider both indoor and outdoor scenarios, using the Stanford Large-Scale 3D Indoor Spaces and Paris-CARLA-3D datasets as reference datasets. We find that random and grid sampling outperform farthest point sampling in terms of segmentation performance, with mean intersection-over-union scores of approximately 0.6, while random sampling displays the fastest execution time. For indoor scenarios, using input 3D point clouds with a small spatial extent (i.e., $1\,\mathrm{m}$) yields the best results. For outdoor scenarios, similar performance is obtained for all tested input extents. In an additional experiment, we evaluate a curvature-weighted sampling approach to test whether geometric features derived from 3D point clouds can guide the selection of more informative input points for deep learning models. However, we find that using curvature as a sampling criterion decreases the segmentation performance, indicating a mismatch between the expected relevance of high-curvature points (e.g., points representing object borders) and the internally learned features of the deep learning models.

1. Introduction

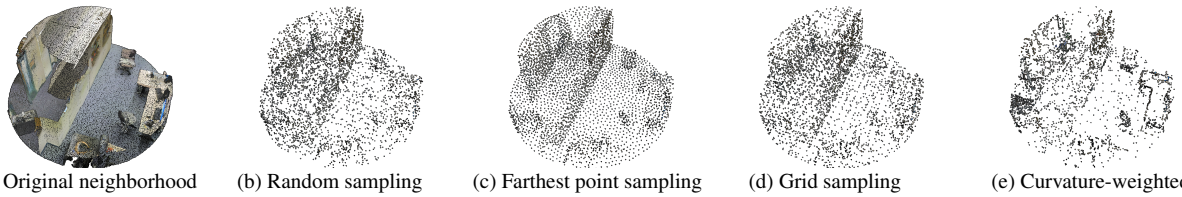
In geospatial applications, 3D point clouds are widely used as point-based 3D models or as base data for 3D model reconstruction. Semantic segmentation, which aims to assign each point in a 3D point cloud to an object category (Xie et al., 2020), plays a fundamental role in applications such as infrastructure management. Over the past decade, deep learning (DL) has achieved outstanding performance in computer vision tasks such as image segmentation; given this success, there is a growing interest in DL approaches for 3D point cloud analysis (Bello et al., 2020; Guo et al., 2021). Furthermore, 3D point clouds captured for geospatial applications typically contain billions of points (Döllner, 2020). Due to computational resource constraints, current DL segmentation approaches rely on dividing the large-scale 3D point clouds into smaller subsets (neighborhood sampling), reducing the density of the 3D point clouds (neighborhood thinning), or a combination of both. We propose the terms *neighborhood sampling* and *neighborhood thinning* in the context of a DL semantic segmentation pipeline, as the terms *subsampling* and *downsampling* are often used interchangeably and inconsistently in the literature.

The size of the objects of interest in geospatial applications can range from the centimeter range (e.g., objects in indoor environments (Armeni et al., 2016)) to several dozens of meters (e.g., assets in outdoor environments (Roynard et al., 2018)). Ideally, the neighborhoods used as input for DL models should be large enough to include any object of interest, and the thinning rate should be low enough to preserve any relevant detail. However, this would result in samples that may still contain hundreds of thousands of points. Processing samples of this size with DL models requires computational resources that are typically unavailable, especially in embedded systems or real-time applications. Furthermore, even current high-end GPUs may not have the memory required to process such volumes of data. Therefore, input sampling for DL models often involves implicit trade-offs between the loss of either contextual or detail information, depending on which approaches are used for neighborhood sampling and thinning. Common approaches for neighborhood thinning are random sampling (RS) (Hu et al., 2020), farthest point sampling (FPS) (Qi et al., 2017b), and grid sampling (GS) (Thomas, 2019). Although these approaches are widely used (Qi et al., 2017a; Yao et al., 2019; Zhang et al., 2019; Wang et al., 2019; Wu et al., 2022), the impact of input sampling techniques on DL-based 3D point cloud semantic segmentation pipelines in terms of the segmentation performance has received little attention so far. In order to contribute to this knowledge gap, this work presents the following contributions:

1. We compare RS, FPS, and GS as neighborhood thinning techniques for DL-based semantic segmentation of 3D point clouds. Our results show that RS and GS have equivalent performance for the downstream segmentation task, while RS displays the faster execution times. In contrast, FPS shows lower segmentation performance and the highest execution time.

2. Inspired by the use of curvature and other surface features derived from 3D point clouds in different sampling and segmentation approaches (Yu et al., 2023; Kumar et al.,

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-4-W6-2025-49-2025 | © Author(s) 2025. CC BY 4.0 License.



(a) Original neighborhood (b) Random sampling (c) Farthest point sampling (d) Grid sampling (e) Curvature-weighted sampling

Figure 1. Output of the neighborhood thinning approaches used in this work for a neighborhood from the Stanford Large-Scale 3D Indoor Spaces Dataset with $3\,\mathrm{m}$ radius and 4096 sampled points.

2019), we investigate if the use of surface features such as curvature as sampling criteria can help retain more informative points for the semantic segmentation task through a custom curvature-weighted sampling (CWS) approach (see Fig. 1). We find that such an approach decreases the performance of the models, pointing to a mismatch between the expected relevance of high-curvature points and the features learned internally by the DL models.

3. We compare the semantic segmentation performance for different spatial extents of the input neighborhoods obtained during neighborhood sampling to gain insight into the trade-off between preserving detail and including scene context information. Our experiments show that, for indoor scenarios, inputs with a smaller spatial extent ($1\,\mathrm{m}$) yield the best results, while similar results are obtained for all tested input extents for outdoor scenarios.

2. Related Work

2.1 Deep Learning-Based Semantic Segmentation of 3D Point Clouds

The goal of semantic segmentation of 3D point clouds is to assign a label to each point, thus separating the point cloud into subsets based on the semantic meanings of its points. Existing DL-based approaches can be divided into projection-, discretization-, and point-based methods (Guo et al., 2021).

2.1.1 Projection-Based Methods These methods generate 2D images from 3D point clouds, thus being able to apply established 2D convolutional neural network (CNN) architectures to them (Bello et al., 2020). Common approaches rely on projecting 3D point clouds into multi-view (Alnaggar et al., 2021; Yang et al., 2020) or spherical images (Cen et al., 2023; Xu et al., 2020). Projection-based methods benefit from using architectures from the mature field of DL-based image segmentation (Bello et al., 2020). However, their performance is sensitive to viewpoint selection and occlusions, and the projection step inevitably introduces information loss (Guo et al., 2021).

2.1.2 Discretization-Based Methods Their goal is to convert 3D point clouds into discrete 3D representations such as voxels and process them with 3D convolutions (Guo et al., 2021). There are approaches based on dense representations that leverage standard 3D convolutions (Zhou and Tuzel, 2018; Rethage et al., 2018) and approaches based on sparse representations that seek to reduce the computation and memory costs of dense CNNs given the spatially-sparse nature of 3D point clouds (Zhao et al., 2022; Yang et al., 2023). Similar to projection-based methods, discretization-based methods build on mature and well-performing CNN architectures. However, the discretization step inherently introduces artifacts and information loss. Furthermore, particularly in dense representations, high resolution translates to high memory and computational costs, while low resolution implies detail loss, making the grid size selection a non-trivial task (Guo et al., 2021; Bello et al., 2020).

2.1.3 Point-Based Methods These methods use 3D point clouds directly as input. However, given the unstructured and unordered nature of 3D point clouds, it is unfeasible to apply standard 3D convolutions. Thus, different approaches to process point clouds directly have been proposed (Guo et al., 2021), including point-wise multi-layer perceptron (MLP)- (Qi et al., 2017b; Hu et al., 2020), convolution- (Thomas et al., 2019; Zhu et al., 2021), graph- (Wang et al., 2019; Lin et al., 2020), transformer- (Wu et al., 2022; Zhang et al., 2022), and hierarchical data structure-based methods (Chen and Wang, 2022; Robert et al., 2023). By using 3D point clouds directly as input, they avoid computational costs associated with converting the point clouds into intermediate representations, as well as conversion artifacts and information loss. However, point clouds do not contain explicit neighboring information. Thus, most point-based methods require computationally expensive neighbor searching strategies, which limit their overall efficiency (Guo et al., 2021).

Most point-based approaches rely on using 3D point cloud thinning strategies such as the ones studied in this work. While this stage introduces information loss similar to that of discretization-based methods, point-based methods are more flexible in terms of the sampling and thinning strategies used in the data processing pipeline (e.g., different geometries, criteria, and resolutions can be used for the selection of input points). Therefore, our work focuses on studying the impact of *neighborhood sampling* and *thinning* techniques (Section 2.2) on the semantic segmentation performance of point-based methods.

2.2 Input Sampling Approaches

Since current DL models cannot consume large-scale 3D point clouds directly, there is a need to sample local neighborhoods from them and to thin these neighborhoods into inputs small enough for the DL models to process (Fig. 2). Given a large-scale input 3D point cloud $\mathcal{P} = \{p_n \in \mathbb{R}^D \mid 1 \leq n \leq N\}$, where D denotes the number of per-point feature dimensions, and N is the total number of points in $\mathcal{P}$, the *neighborhood sampling* stage samples a set of local neighborhoods $\{\mathcal{Q}_s \subseteq \mathcal{P} \mid 1 \leq s \leq S\}$ from $\mathcal{P}$, where S denotes the number of sampled local neighborhoods. In the *neighborhood thinning* stage, the number of points in a local neighborhood $\mathcal{Q}_s$ is reduced, often by sampling a fixed number of points (Wang et al., 2019; Zhang et al., 2019) (typically, a few thousand points). Approaches for the neighborhood sampling step include the use of different

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-X-4-W6-2025-49-2025 | © Author(s) 2025. CC BY 4.0 License.

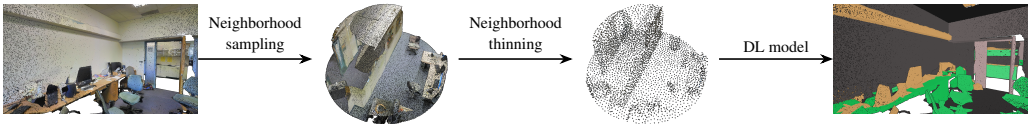


Figure 2. Data processing pipeline for DL-based semantic segmentation of large-scale 3D point clouds as used in this work.

neighborhood geometries such as spheres (Thomas, 2019), cylinders (Xiang et al., 2023), and cubes (Wang et al., 2019). For the neighborhood thinning step, heuristic approaches such as RS, FPS, and GS are commonly used. RS randomly selects K points from each element in $\mathcal{Q}_s$. FPS samples K points from each element in $\mathcal{Q}_s$ through a re-ordering of the metric space $\{p_1 \cdots p_k \cdots p_K\}$ such that each p_k is the point that is the farthest from the first $k-1$ points. Compared to RS, FPS provides a better coverage of the entire point set (Yao et al., 2019). The GS strategy selects K points from each element in $\mathcal{Q}_s$ by partitioning the 3D space into 3D voxels of a given size and selecting one point in each occupied one (Dinesh et al., 2020). In contrast with RS and FPS, GS does not return a fixed number of points. Among the works that use RS for thinning DL input data are PointNet (Qi et al., 2017a), PointNet++ (Qi et al., 2017b), ShellNet (Zhang et al., 2019) and Dynamic Graph CNN (DGCNN) (Wang et al., 2019). GS is used for input thinning in KP-FCNN (Thomas et al., 2019), Point Transformer (Zhao et al., 2021) and Point Transformer V2 (PTv2) (Wu et al., 2022).

In addition to thinning the input data, most DL architectures for 3D point cloud semantic segmentation follow an encoder-decoder scheme, where the input 3D point clouds are further thinned in the encoder layers (i.e., pooling operations). PointNet++ (Yao et al., 2019) uses FPS, KP-FCNN (Thomas et al., 2019) uses GS, and RandLA-Net (Hu et al., 2020) uses RS to gradually thin 3D point clouds in the encoder layers. Currently, there is more attention in the literature on studying and comparing these internal thinning approaches of the models. For instance, Hu et al. (2020) find that RS provides an advantage when compared to other heuristic and learning-based approaches in terms of execution time and memory consumption. In contrast, few works analyze the impact of input sampling approaches on the segmentation performance. For example, Ma et al. (2020) look at the input 3D point clouds and analyze the impact of the neighborhood thinning step on the downstream segmentation task. However, they focus on a specific indoor application and only evaluate the impact of different numbers of K input points through the use of RS, and do not consider other sampling techniques. Pierdicca et al. (2020) evaluate the use of RS, GS, and octree-based sampling for the semantic segmentation of cultural heritage sites. However, the evaluation criteria for the sampling techniques refer to their practicality of use and not to their impact on the segmentation performance. Grandio et al. (2022) evaluate the use of GS as a neighborhood thinning strategy for the DL-based segmentation of railway environments. Although the authors study the impact of the grid size on the segmentation performance, they focus on a specific task, and do not compare it against other thinning strategies. Deschaud et al. (2021) evaluate the impact of different neighborhood sampling radius sizes on the semantic segmentation task; however, they focus on outdoor scenarios and do not evaluate the use of different neighborhood thinning approaches.

In our work, we focus on heuristic techniques, as they are widely used in the neighborhood thinning stage of DL-based 3D point cloud semantic segmentation pipelines (Qi et al., 2017a; Yao et al., 2019; Zhang et al., 2019; Wang et al., 2019; Zhao et al., 2021; Wu et al., 2022), are model- and task-agnostic, and are easily adapted to different semantic segmentation pipelines. In contrast, learning-based approaches require further training, present a higher implementation complexity, and are oriented for a specific task (Liu et al., 2022; Wang and Zhao, 2023). Furthermore, learning-based approaches are often structured as internal model layers or modules (Wu et al., 2023; Chen et al., 2023) and cannot be directly adapted as a stage of the data processing pipeline. Some approaches require pre-trained models (Dovrat et al., 2019), for which heuristic sampling techniques would still be needed. Thus, we evaluate four heuristic neighborhood thinning strategies, i.e., RS, FPS, GS, and CWS (Section 3.2) for input neighborhoods of varying spatial extents and three widely used DL architectures, considering outdoor and indoor application scenarios.

Surface Features for 3D Point Cloud Thinning The extraction of feature lines, i.e., ordered connections of feature points, is a vital operation for the processing and understanding of 3D models (Nie, 2016). These feature points represent surface variations such as creases, borders, and corners (Gumhold et al., 2001) and can be used to abstract complex 3D shapes, thus facilitating tasks such as surface reconstruction and shape classification (Zhu et al., 2023). Hence, feature points have the potential to provide relevant information for DL models used in 3D point cloud analysis tasks such as semantic segmentation, as the preservation of these feature points during the neighborhood thinning stage could be of considerable value for the downstream segmentation task. Although there is work related to feature-based 3D point cloud thinning using heuristic (Yu et al., 2023) or learning-based techniques (Ye et al., 2022), approaches such as RS and GS remain as some of the most widely used in DL-based 3D point cloud semantic segmentation pipelines. To our knowledge, the use of feature-based sampling techniques for the neighborhood thinning stage and its impact on the segmentation performance has not yet been studied in detail.

3. Data and Methods

We use the Paris-CARLA-3D dataset (Deschaud et al., 2021) for the segmentation of outdoor scenes and the Stanford Large-Scale 3D Indoor Spaces Dataset (S3DIS) dataset (Armeni et al., 2016) for indoor scene segmentation. Appendix A introduces both datasets in more detail, and shows the training, validation, and testing partitions that we use for the present study.

3.1 Data Preprocessing

Following a similar approach as Thomas et al. (2019), we initially reduce the point density of the S3DIS and PARIS datasets through grid sampling. We set the grid sizes to $2\,\mathrm{cm}$ and $6\,\mathrm{cm}$, respectively, to speed up subsequent processing steps while preserving sufficient detail for the segmentation tasks. These val-

ues are selected taking those in (Thomas et al., 2019) as reference. We use point coordinates and RGB values as model input for both datasets. Similar to the work by Kumar et al. (2019), in which the authors find that using surface information as additional per-point features improves the segmentation performance, we calculate normal vectors and curvature values for each point and add them to the input features. The normal vector of a point is estimated by calculating the eigenvectors of the 3D covariance matrix of a point's k_n nearest neighbors. The eigenvector with the smallest eigenvalue is taken as the normal vector (Hoppe et al., 1992). To calculate the curvature value of a point, a tangent plane is spanned by the point and its normal vector. The curvature value is defined as the average distance of the point's k_c nearest neighbors to this tangent plane (Pauly et al., 2002). In this work, we use empirically selected values of $k_n = 78$ and $k_c = 16$. The 3D point clouds in the PARIS dataset contain some outliers that produce undesirably high curvature values. We manually remove these outliers using the CloudCompare software.[1]

3.2 Sampling of Model Inputs

As shown in Fig. 2, we consider a two-stage sampling procedure: In the first stage (neighborhood sampling), a set of local neighborhoods $\mathcal{Q}_s$ with a fixed spatial extent is sampled from the large-scale 3D point cloud. In the second stage (neighborhood thinning), a fixed number of K points is sampled from each neighborhood (Section 2.2). We set the requirement that each neighborhood should contain a fixed number of points, as this simplifies the batch processing of the data and ensures that the DL model can process a batch of samples with a given GPU memory budget. Following the approach of Thomas et al. (2019), we use spherical neighborhoods with a fixed radius r for the neighborhood sampling stage. These neighborhoods are obtained by sampling S center points $c_1, ..., c_S$ from $\mathcal{P}$ and searching all points within radius r around these center points:

$$\mathcal{Q}_s = \{p \in \mathcal{P} \mid \|p - c_s\| \leq r\}, \tag{1}$$

where $\|...\|$ denotes the 3D Euclidean distance between two points. We experiment with different values for r, namely 1, 3, and $6\,\mathrm{m}$ for the S3DIS dataset and 3, 6, and $9\,\mathrm{m}$ for the PARIS dataset. To mitigate imbalances in the class distribution, we weight the sampling of center points c_s by their semantic class label during training. The sampling probability of a point p_n is set proportional to the inverse class frequency $\frac{1}{N_c}$, where N_c denotes the number of points with the same semantic class label as p_n across all training 3D point clouds. During validation and inference, we uniformly sample the center points without considering their semantic class labels.

In the neighborhood thinning stage, a fixed number of points is sampled from each neighborhood $\mathcal{Q}_s$. For this purpose, we consider RS, GS, and FPS. In line with (Wang et al., 2019), we set the number of points sampled from each neighborhood to 4096. If a neighborhood $\mathcal{Q}_s$ contains less than 4096 points, we randomly duplicate points from $\mathcal{Q}_s$. For GS, the grid size is set to $0.03\,\mathrm{m}$ for the S3DIS dataset and to $0.08\,\mathrm{m}$ for the PARIS dataset ensuring to have larger grid sizes than in the preprocessing stage and thus avoid having too many empty voxels. We randomly sample additional points when GS retains less than the desired 4096 and randomly discard points when GS retains too many points. In addition to the aforementioned sampling algorithms, we evaluate a CWS algorithm to evaluate the feasibility of using surface features derived from 3D point clouds as a criterion for neighborhood thinning. Our sampling algorithm works as follows: To sample K points from a 3D point cloud $\mathcal{Q}_s$, we select the $\lfloor 0.7 \cdot K \rfloor$ points with the highest curvature values from $\mathcal{Q}_s$ and augment them with $\lceil 0.3 \cdot K \rceil$ points randomly sampled from $\mathcal{Q}_s$. In this way, we aim to preserve more details for areas with strong surface variation, which are characterized by high curvature values. By randomly sampling $30\,\%$ of the points, we aim to cover smooth surfaces with low curvature values, albeit with a lower point density.

[1] https://cloudcompare.org/

3.3 Experimental Setup

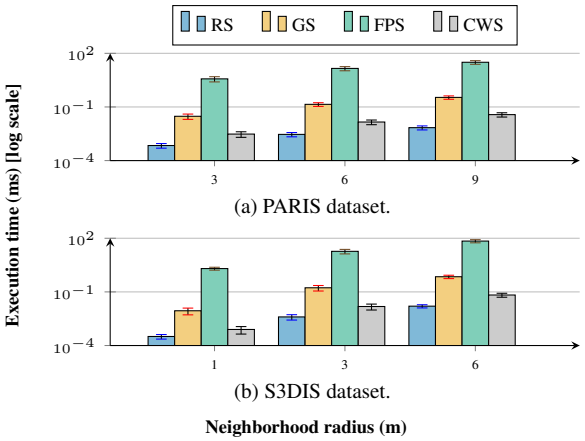
We compare the execution times of the four analyzed thinning techniques (RS, FPS, GS, and CWS) for the different neighborhood radius sizes specified in Section 3.2. We randomly sample ten neighborhoods of each radius size from each dataset and perform the execution time benchmarking ten times per neighborhood, thinning the neighborhoods to 4096 points for all sampling methods. To evaluate the sampling and thinning techniques in terms of downstream semantic segmentation performance, we use the DGCNN (Wang et al., 2019), RandLA-Net (Hu et al., 2020), and PTv2 (Wu et al., 2022) architectures, as they represent graph-, MLP-, and transformer-based approaches, respectively. Given the considerably high execution times of the FPS approach (Fig. 3), we limit the evaluation of FPS to the $1\,\mathrm{m}$ and $3\,\mathrm{m}$ radii of the S3DIS and PARIS datasets, respectively, as evaluating larger neighborhoods that contain a higher amount of points was not feasible. To assess the segmentation performance, we use the mean intersection-over-union (mIoU) metric. In Appendix B of the supplementary material we provide details concerning the used architectures, hyperparameters, the mIoU calculation during training, validation, and testing, and additional implementation details.

4. Results

Although FPS provides visually pleasing results (Fig. 1c), it has the slowest execution time compared to the other approaches (Fig. 3). Meanwhile, RS provides the fastest sampling speed, thus making it desirable as a neighborhood thinning strategy. However, from a visual perspective, objects and shapes are often more difficult to recognize in the thinned neighborhoods (Fig. 1b). CWS and GS have higher execution times than RS but are still considerably faster than FPS. As shapes and objects are visually more recognizable in the FPS, GS, and curvature-weighted results than with RS, this makes them worth considering as neighborhood thinning strategies. Thus, we compare the semantic segmentation performance when using these approaches to provide a basis for selecting a thinning strategy.

Fig. 4 shows the results on the test set of the PARIS dataset. Overall, GS and RS produce similar IoU scores for most semantic classes and radius sizes, of approximately 0.6. The curvature-weighted approach is outperformed for most classes by RS and GS. For the $3\,\mathrm{m}$ radius, FPS displays a lower performance than RS and GS. Similar mIoU values are obtained for all radii, with the best values being achieved for $6\,\mathrm{m}$ by a small margin.

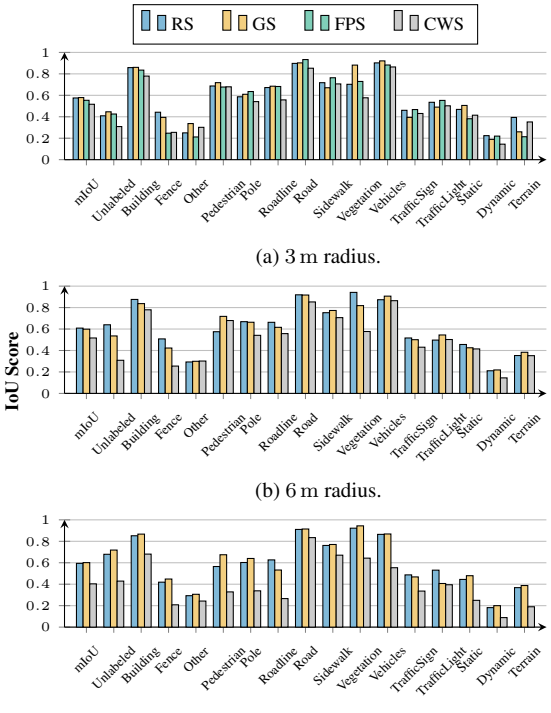
Fig. 5 shows the results for the test set of the S3DIS dataset. Overall, RS and GS display similar results, with RS showing a slight advantage on bigger radius sizes. In this case, a decrease in the segmentation performance can be observed for



(a) PARIS dataset.

(b) S3DIS dataset.

Figure 3. Execution times of different neighborhood thinning algorithms on both datasets. The vertical lines on top of the bars represent the standard deviation.

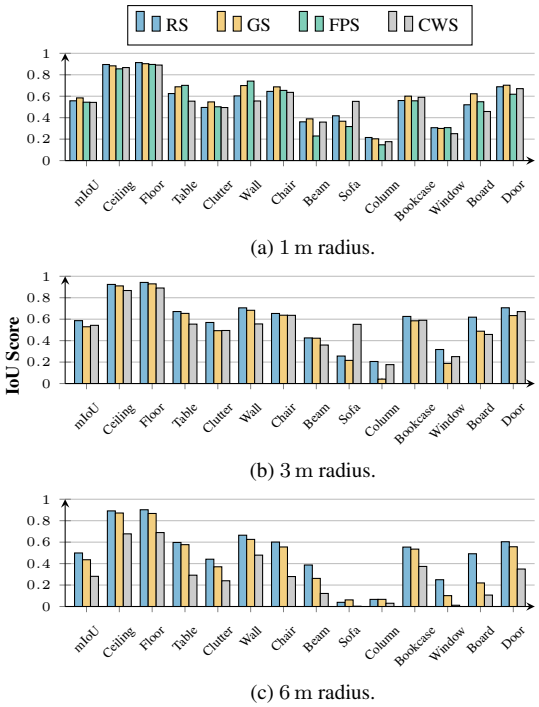


(a) 3 m radius.

(b) 6 m radius.

(c) 9 m radius.

Figure 4. IoU scores for the test set of the PARIS dataset.



(a) 1 m radius.

(b) 3 m radius.

(c) 6 m radius.

Figure 5. IoU scores for the test set of the S3DIS dataset.

most objects as the radius size increases. Furthermore, with the only exception of the *sofa* class on the radius size of 1 m, the curvature-weighted approach is outperformed by both RS and GS. For the 1 m radius (Fig. 5a), FPS displays an overall performance similar to the curvature-weighted approach.

Table 1 shows the condensed per-architecture results for both datasets. The PTv2 architecture shows the best overall segmentation results. For PTv2 and RandLA-Net, RS yields slightly better mIoU scores than GS for most radius sizes on the PARIS dataset. For DGCNN, the GS strategy shows a slight advantage for the PARIS dataset for all radii. In contrast, for the S3DIS dataset, the RS approach produces slightly better mIoU scores than GS in most cases. For the radius size of 1 m and the PTv2 architecture, FPS shows performance on par with GS. Additionally, RandLA-Net shows the lowest results for the curvature-weighted approach. Appendix C of the supplementary material provides visualizations of the training and validation results, the per-architecture test results, and the full experimental results.

Architecture	Strategy	PARIS			S3DIS		
		3 m	6 m	9 m	1 m	3 m	6 m
DGCNN	RS	0.55	0.60	0.61	**0.59**	**0.55**	**0.50**
	FPS	0.57	-	-	0.53	-	-
	GS	**0.58**	**0.63**	**0.62**	0.58	0.53	0.38
	CWS	0.56	0.45	0.45	0.55	0.42	0.33
RandLA-Net	RS	0.53	**0.58**	**0.55**	0.42	**0.53**	**0.45**
	FPS	**0.56**	-	-	0.41	-	-
	GS	**0.56**	0.53	0.54	**0.48**	0.49	0.40
	CWS	0.46	0.40	0.26	0.42	0.41	0.31
PTv2	RS	**0.65**	**0.65**	0.62	0.67	**0.67**	**0.55**
	FPS	0.53	-	-	**0.70**	-	-
	GS	0.61	0.64	**0.65**	**0.70**	0.57	0.52
	CWS	0.53	0.53	0.50	0.66	0.33	0.20

Table 1. Per-architecture semantic segmentation mIoU scores for the test set of the PARIS and S3DIS datasets.

5. Discussion

Execution Times and Semantic Segmentation Performance Our results show no clear difference in terms of the downstream semantic segmentation performance between the RS and GS approaches. However, RS shows faster execution times (Fig. 3) and is therefore preferable. FPS presented no substantial benefit for the downstream segmentation task, only displaying a performance on par with RS for PTv2 on the S3DIS dataset and RandLA-Net on the PARIS dataset while requiring significantly higher execution times (Fig. 3 and Table 1). Therefore, it was unfeasible for sampling larger neighborhoods in the context of the present work. These results corroborate the findings by Hu (2024) on the efficiency of RS. The curvature-weighted approach did not provide an increase in the semantic segmentation performance. Nevertheless, given its lower execution time when compared to GS and FPS (Fig. 3) and the overall low segmentation scores (rarely surpassing 0.6), the investigation of feature-based approaches able to extract more informative fea-

tures stands as a potential avenue for further research. However, the time required to compute per-point features should also be taken into account when designing DL-based segmentation pipelines. In our work, the curvature-weighted approach relies on pre-computed curvature values generated during data preprocessing (see Section 3.1). As these values only need to be calculated once, this execution time was not considered for the benchmarking shown in Fig. 3.

Semantic Segmentation Performance Across Neighborhood Thinning Approaches Across most classes in both datasets, FPS and the curvature-weighted approach are consistently outperformed by both GS and RS (Section 4 and Appendix C). These results are counterintuitive in that it would be reasonable to assume that, for FPS, having a better coverage of the entire point set would benefit the DL model for the correct identification of objects. For CWS, feature points, i.e., points that delineate surface elements such as creases and borders, would be expected to be more informative than points from flat, uniform surfaces. This behavior could be due to several factors: First, in our pipeline, the random seed is increased by 1 after each training epoch. This means that RS outputs a different subset of points from those neighborhoods each epoch, creating a data augmentation effect that potentially helps the models learn to generalize over a wider array of inputs. Since our implementation of GS randomly adds and discards points from the output to achieve the required input size (Section 3.2 and Appendix B.5.1), it is likely that a similar data augmentation effect is present. In contrast, this effect is not present in FPS, whereas it is considerably diminished in the curvature-weighted approach, since the curvature values are pre-computed for the whole large-scale point cloud. Second, the use of FPS as a neighborhood thinning strategy is based on the underlying assumption that DL semantic segmentation models benefit from the uniform coverage of the point set it provides; i.e., that all regions within the neighborhood are equally important to the models. In contrast, RS has the inherent bias of sampling more points from high-density areas, which potentially contain more meaningful geometric relationships than those in sparse areas. These results suggest that not all regions and points in a 3D point cloud are of equal relevance for the evaluated models. Third, following this hypothesis, the curvature-weighted approach assumes that points with high curvature values provide more valuable information to the DL models than points located in relatively flat surfaces. However, our results suggest that this is not the case, pointing to a mismatch between the expected relevance of high curvature points and the internally learned features of the deep learning models. Further research on ablation studies on hyperparameters such as k_n, k_c for normal and curvature computation, as well as the split percentages in CWS, and the exploration of different feature combinations (e.g., normals and geometric complexity), could shed light on the apparent failure of geometric priors in the context of the present work. It is possible that non-geometric features such as color or intensity contrast could provide more information. However, due to the black-box nature of DL models, it is difficult to ascertain which features are internally learned by the models and, thus, which points would be the most relevant to prioritize during the neighborhood thinning step. The development of explainable AI (XAI) techniques for semantic segmentation DL models could improve the current understanding of the informative value of different input points.

Per-Architecture Results When considering the per-architecture results, no clear difference between the performance of RS, FPS, and GS can be identified, as shown in Table 1 and Appendix C. These results suggest that neighborhood thinning approaches are agnostic to the DL architecture of the downstream segmentation task. A larger study covering a wider range of architectures would further help ascertain the generalizability of these results.

Neighborhood Extent Regarding the spatial extent of the neighborhoods, we find that, for outdoor environments, larger neighborhoods tend to produce better results, while the opposite tendency can be observed for indoor environments (Table 1, Figs. 4 and 5, and Appendix C). On the S3DIS dataset, we observe a decrease in performance as the neighborhood radius increases (Section 4, Table 1, and Appendix C). This behavior is likely due to the size of the objects to be analyzed: Since indoor objects are typically no larger than a few meters in size, the loss of detail information introduced by larger sampling radii presumably does not outweigh the benefit of including more context information. In contrast, larger radius sizes produce slightly better results for the PARIS dataset. Given that outdoor structures are typically larger, context information captured by using larger radii seems to be more relevant than the fine-grained details for outdoor scenarios.

Statistical Significance We use a linear mixed-effects model to study the individual effect of the sampling approaches, architectures, and sampling radii on the segmentation performance. We set these three variables as fixed effects and included the semantic class as a random effect to account for the variability in IoU scores across classes. For both datasets, the null hypothesis is that a given parameter (i.e., an architecture, sampling strategy, or sampling radius) has no effect on the mIoU. Table 2 and Table 3 in Appendix C.1 show the full model results for the PARIS and S3DIS datasets, respectively. For both datasets, taking CWS as a baseline, we observe a slight increase in the mIoU for FPS, and a larger one for GS and RS, in line with our observations. Furthermore, we observe p-values lower than 0.01, allowing us to reject the null hypothesis for the sampling strategies. Regarding the architecture, taking DGCNN as a baseline, we observe a decrease in the mIoU for RandLA-Net and an increase for PTv2 - in line with our observations. However, for the PARIS dataset, we observe a p-value > 0.01 for PTv2, indicating that we cannot reject the null hypothesis for this architecture and suggesting that further experiments are needed to fully ascertain the benefit of PTv2 for outdoor environments. Regarding the sampling radius, the results for the S3DIS dataset show a decrease in the mIoU when the radius size increases, supporting our conclusions and showing a p-value < 0.01. For the PARIS dataset, the change in mIoU is very small (showing a coefficient of -0.004) when increasing the radius size; however, we observe a p-value > 0.01, indicating the need for further experimentation to fully ascertain the effect of the radius size on outdoor environments.

Known Limitations Additional factors could impact the segmentation results. For instance, neighborhoods with a higher number of points, although more memory expensive, could provide better segmentation results across different sampling approaches. Furthermore, the use of different geometries for neighborhood sampling might affect the results as well, as different geometries (e.g., cubes and cylinders) might be better suited to capture different objects in their entirety (e.g., trees and buildings), potentially increasing the segmentation performance as well. Another potential limitation is that the PARIS dataset covers a relatively small area when compared with other

outdoor datasets (e.g., SemanticKITTI (Behley et al., 2019)). DL models might benefit from training on such datasets, as they contain more instances and variations of the object classes to be segmented. An additional factor that may influence the segmentation results is the internal receptive fields of the different DL architectures. Depending on the internal sampling resolution and approach of a DL model, the preservation of a higher resolution or specific points of interest might have a diminished effect on the segmentation performance. The impact and interaction of these factors and additional ones, such as class imbalance, on the segmentation task requires further investigation. Finally, benchmarking a wider array of point-based architectures and including discretization- and projection-based techniques is a potential avenue for future work.

6. Conclusions

Through this study, we show that the selection of sampling technique for neighborhood thinning in DL-based semantic segmentation pipelines can have a noticeable impact in the downstream segmentation task, in terms of both the segmentation performance and the overall execution time. Our results indicate that RS provides more benefits than other, more specialized heuristics in terms of both segmentation performance and scalability to larger 3D point clouds - both vital factors in geospatial applications. Furthermore, the selection of an appropriate spatial extent for the sampling of local neighborhoods is shown to be of particular relevance for indoor environments, where too big a sampling area might cause the loss of relevant detail information. These results provide practicable guidelines for the development of faster and better-performing DL-based semantic segmentation pipelines. Furthermore, our results underline the importance of the development of XAI techniques that can improve our understanding of the point features that are informative for the DL models, allowing for the development of better heuristic approaches.

Acknowledgements

This work was partially funded through grants by the Systems Design Research School of the Hasso Plattner Institute and by the Federal Ministry of Education and Research, Germany through grant 033L305A (‘TreeDigitalTwins’) and grant 01IS22062 (AI research group ‘FFS-AI’). We thank the anonymous reviewers for their valuable feedback.

References

Alnaggar, Y. A., Afifi, M., Amer, K., ElHelw, M., 2021. Multi projection fusion for real-time semantic segmentation of 3d lidar point clouds. *WACV*, 1800–1809.

Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S., 2016. 3d semantic parsing of large-scale indoor spaces. *CVPR*, IEEE, 1534–1543.

Behley, J., Garbade, M., Milioto, A., Quenzel, J., Behnke, S., Stachniss, C., Gall, J., 2019. Semantickitti: A dataset for semantic scene understanding of lidar sequences. *ICCV*, IEEE, 9296–9306.

Bello, S. A., Yu, S., Wang, C., Adam, J. M., Li, J., 2020. Review: Deep Learning on 3D Point Clouds. *Remote Sensing*, 12(11), 1729.

Bengio, Y., LeCun, Y. (eds), 2015. Adam: A Method for Stochastic Optimization.

Cen, J., Zhang, S., Pei, Y., Li, K., Zheng, H., Luo, M., Zhang, Y., Chen, Q., 2023. Cmdfusion: Bidirectional fusion network with cross-modality knowledge distillation for lidar semantic segmentation.

Chen, C., Yuan, H., Liu, H., Hou, J., Hamzaoui, R., 2023. Casnet: Cascade attention-based sampling neural network for point cloud simplification. *ICME*, 1991–1996.

Chen, J.-K., Wang, Y.-X., 2022. Pointtree: Transformation-robust point cloud encoder with relaxed k-d trees. S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, T. Hassner (eds), *Computer Vision – ECCV 2022*, Springer Nature Switzerland, Cham, 105–120.

Deschaud, J.-E., Duque, D., Richa, J. P., Velasco-Forero, S., Marcotegui, B., Goulette, F., 2021. Paris-CARLA-3D: A Real and Synthetic Outdoor Point Cloud Dataset for Challenging Tasks in 3D Mapping. *Remote Sensing*, 13(22).

Dinesh, C., Cheung, G., Wang, F., Bajić, I. V., 2020. Sampling of 3d point cloud via gershgorin disc alignment. *ICIP*, 2736–2740.

Döllner, J., 2020. Geospatial Artificial Intelligence: Potentials of Machine Learning for 3D Point Clouds and Geospatial Digital Twins. *PFG – J. of Photogrammetry, Remote Sensing and Geoinformation Sci.*, 88(1), 15-24.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., Koltun, V., 2017. CARLA: An open urban driving simulator. S. Levine, V. Vanhoucke, K. Goldberg (eds), *CoRL*, Proceedings of Machine Learning Res., 78, PMLR, 1–16.

Dovrat, O., Lang, I., Avidan, S., 2019. Learning to Sample. *CVPR*, 2760–2769.

Grandio, J., Riveiro, B., Soilán, M., Arias, P., 2022. Point cloud semantic segmentation of complex railway environments using deep learning. *Automation in Construction*, 141, 104425.

Gumhold, S., Wang, X., Macleod, R., 2001. Feature extraction from point clouds. *SIAM IMR*.

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M., 2021. Deep Learning for 3D Point Clouds: A Survey. *IEEE Trans. on Pattern Analysis and Machine Intell.*, 43(12), 4338-4364.

Guo, Z., Feng, C.-C., 2020. Using multi-scale and hierarchical deep convolutional features for 3D semantic classification of TLS point clouds. *Int. J. of Geographical Inf. Sci.*, 34(4), 661-680.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., Stuetzle, W., 1992. Surface reconstruction from unorganized points. *SIGGRAPH*, SIGGRAPH ’92, Association for Computing Machinery, New York, NY, USA, 71–78.

Hu, Q., 2024. Sampling Strategies for Efficient Segmentation and Object Detection of 3D Point Clouds. *World Scientific Annu. Rev. of Artif. Intell.*, 02, 2440007.

Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A., 2020. RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds. *CVPR*.

Kumar, A., Anders, K., Winiwarter, L., Höfle, B., 2019. Feature Relevance Analysis for 3D Point Cloud Classification using Deep Learning. *ISPRS Ann. of the Photogrammetry, Remote Sensing and Spatial Inf. Sciences*, IV-2/W5, 373–380.

Lin, Z.-H., Huang, S.-Y., Wang, Y.-C. F., 2020. Convolution in the cloud: Learning deformable kernels in 3d graph convolution networks for point cloud analysis. *CVPR*.

Liu, J., Guo, J., Xu, D., 2022. APSNet: Toward Adaptive Point Sampling for Efficient 3D Action Recognition. *IEEE Trans. on Image Processing*, 31, 5287-5302.

Ma, J. W., Czerniawski, T., Leite, F., 2020. Semantic segmentation of point clouds of building interiors with deep learning: Augmenting training datasets with synthetic BIM-based point clouds. *Automation in Construction*, 113, 103144.

Nie, J., 2016. Extracting feature lines from point clouds based on smooth shrink and iterative thinning. *Graphical Models*, 84, 38-49.

Pauly, M., Gross, M., Kobbelt, L., 2002. Efficient simplification of point-sampled surfaces. *IEEE Visualization, 2002. VIS 2002.*, 163–170.

Pierdicca, R., Paolanti, M., Matrone, F., Martini, M., Morbidoni, C., Malinverni, E. S., Frontoni, E., Lingua, A. M., 2020. Point Cloud Semantic Segmentation Using a Deep Learning Framework for Cultural Heritage. *Remote Sensing*, 12(6).

Qi, C. R., Su, H., Mo, K., Guibas, L. J., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. *CVPR*, IEEE, 77–85.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017b. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *arXiv preprint arXiv:1706.02413*.

Rethage, D., Wald, J., Sturm, J., Navab, N., Tombari, F., 2018. Fully-convolutional point networks for large-scale point clouds. V. Ferrari, M. Hebert, C. Sminchisescu, Y. Weiss (eds), *ECCV*, Lecture Notes in Computer Sci., 11208, Springer Int., 625–640.

Robert, D., Raguet, H., Landrieu, L., 2023. Efficient 3D Semantic Segmentation with Superpoint Transformer. *ICCV*.

Roynard, X., Deschaud, J.-E., Goulette, F., 2018. Paris-Lille-3D: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification. *The Int. J. of Robot. Res.*, 37(6), 545-557.

Thomas, H., 2019. Learning new representations for 3D point cloud semantic segmentation. Theses, Université Paris sciences et lettres.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L., 2019. Kpconv: Flexible and deformable convolution for point clouds. *ICCV*, IEEE, 6410–6419.

Wang, Q., Ma, Y., Zhao, K., Tian, Y., 2022. A Comprehensive Survey of Loss Functions in Machine Learning. *Ann. of Data Sci.*, 9(2), 187–212.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., Solomon, J. M., 2019. Dynamic Graph CNN for Learning on Point Clouds. *ACM Trans. on Graph. (TOG)*.

Wang, Y., Zhao, L., 2023. Point cloud sampling method based on offset-attention and mutual supervision. *The Visual Computer*, 39(6), 2337-2345.

Wu, C., Zheng, J., Pfrommer, J., Beyerer, J., 2023. Attention-based point cloud edge sampling. *CVPR*, 5333–5343.

Wu, X., Lao, Y., Jiang, L., Liu, X., Zhao, H., 2022. Point transformer v2: Grouped vector attention and partition-based pooling. *NeurIPS*.

Xiang, B., Peters, T., Kontogianni, T., Vetterli, F., Puliti, S., Astrup, R., Schindler, K., 2023. Towards accurate instance segmentation in large-scale lidar point clouds.

Xie, Y., Tian, J., Zhu, X. X., 2020. Linking Points With Labels in 3D: A Review of Point Cloud Semantic Segmentation. 8(4), 38-59.

Xu, C., Wu, B., Wang, Z., Zhan, W., Vajda, P., Keutzer, K., Tomizuka, M., 2020. Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation. A. Vedaldi, H. Bischof, T. Brox, J.-M. Frahm (eds), *Computer Vision – ECCV 2020*, Springer Int. Publishing, Cham, 1–19.

Yang, J., Lee, C., Ahn, P., Lee, H., Yi, E., Kim, J., 2020. Pbp-net: Point projection and back-projection network for 3d point cloud segmentation. *IROS*, 8469–8475.

Yang, Y.-Q., Guo, Y.-X., Xiong, J.-Y., Liu, Y., Pan, H., Wang, P.-S., Tong, X., Guo, B., 2023. Swin3d: A pretrained transformer backbone for 3d indoor scene understanding.

Yao, X., Guo, J., Hu, J., Cao, Q., 2019. Using Deep Learning in Semantic Classification for Point Cloud Data. *IEEE Access*, 7, 37121-37130.

Ye, Y., Yang, X., Ji, S., 2022. APSNet: Attention Based Point Cloud Sampling. *BMVC*.

Yu, J., Wu, G., Wu, W., Ma, W., Chang, H., Wei, Z., Jiang, X., Xu, J., 2023. Construction quality detection based on point cloud nonuniform thinning method. *Structures*, 56, 104930.

Zhang, J., Li, X., Zhao, X., Zhang, Z., 2022. LLGF-Net: Learning Local and Global Feature Fusion for 3D Point Cloud Semantic Segmentation. *Electronics*, 11(14).

Zhang, Z., Hua, B.-S., Yeung, S.-K., 2019. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics. *ICCV*.

Zhao, H., Jiang, L., Jia, J., Torr, P. H., Koltun, V., 2021. Point transformer. *ICCV*, 16259–16268.

Zhao, L., Xu, S., Liu, L., Ming, D., Tao, W., 2022. SVASeg: Sparse Voxel-Based Attention for 3D LiDAR Point Cloud Semantic Segmentation. *Remote Sensing*, 14(18).

Zhou, Y., Tuzel, O., 2018. Voxelnet: End-to-end learning for point cloud based 3d object detection. *CVPR*.

Zhu, X., Du, D., Chen, W., Zhao, Z., Nie, Y., Han, X., 2023. Nerve: Neural volumetric edges for parametric curve extraction from point cloud. *CVPR*, 13601–13610.

Zhu, X., Zhou, H., Wang, T., Hong, F., Ma, Y., Li, W., Li, H., Lin, D., 2021. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. *CVPR*, 9939–9948.