sensors

MDPI

Article

TransNet: Transformer-Based Point Cloud Sampling Network

Hookyung Lee [1], Jaeseung Jeon [1], Seokjin Hong [1], Jeesu Kim [2] and Jinwoo Yoo [3,*]

[1] Graduate School of Automotive Engineering, Kookmin University, Seoul 02707, Republic of Korea; gnrud099@gmail.com (H.L.); sing5386@naver.com (J.J.); cheongsu030536@gmail.com (S.H.)
[2] Departments of Cogno-Mechatronics Engineering and Optics and Mechatronics Engineering, Pusan National University, Busan 46241, Republic of Korea; jeesukim@pusan.ac.kr
[3] Department of Automobile and IT Convergence, Kookmin University, Seoul 02707, Republic of Korea
* Correspondence: jwyoo@kookmin.ac.kr

Abstract: As interest in point cloud processing has gradually increased in the industry, point cloud sampling techniques have been researched to improve deep learning networks. As many conventional models use point clouds directly, the consideration of computational complexity has become critical for practicality. One of the representative ways to decrease computations is downsampling, which also affects the performance in terms of precision. Existing classic sampling methods have adopted a standardized way regardless of the task-model property in learning. However, this limits the improvement of the point cloud sampling network's performance. That is, the performance of such task-agnostic methods is too low when the sampling ratio is high. Therefore, this paper proposes a novel downsampling model based on the transformer-based point cloud sampling network (TransNet) to efficiently perform downsampling tasks. The proposed TransNet utilizes self-attention and fully connected layers to extract meaningful features from input sequences and perform downsampling. By introducing attention techniques into downsampling, the proposed network can learn about the relationships between point clouds and generate a task-oriented sampling methodology. The proposed TransNet outperforms several state-of-the-art models in terms of accuracy. It has a particular advantage in generating points from sparse data when the sampling ratio is high. We expect that our approach can provide a promising solution for downsampling tasks in various point cloud applications.

Keywords: deep learning; transformer; self-attention; multi-head attention; point cloud; down sampling; classification; network

Citation: Lee, H.; Jeon, J.; Hong, S.; Kim, J.; Yoo, J. TransNet: Transformer-Based Point Cloud Sampling Network. *Sensors* **2023**, *23*, 4675. https://doi.org/10.3390/s23104675

Academic Editors: Shyan-Ming Yuan, Zeng-Wei Hong and Wai-Khuen Cheng

Received: 11 April 2023
Revised: 9 May 2023
Accepted: 9 May 2023
Published: 11 May 2023

1. Introduction

The technology for creating point clouds using 3D sensings, such as RGB-D cameras and LiDAR, is advancing rapidly, and increases in computing speeds and interest in the 3D point cloud field are drawing attention as well [1–3]. This has raised the importance of point clouds in various fields. Point clouds provide a detailed and accurate representation of real-world objects and environments, allowing for the precise measurements, analysis, and manipulation of 3D data. They have numerous applications in fields such as robotics, autonomous vehicles, virtual reality, architecture, and cultural heritage preservation. As the technology for creating point clouds continues to improve, we can anticipate an even greater reliance on these data structures for a wide range of applications.

Because the form of 3D point cloud data differs from that of a typical image or natural language processing (NLP) data, when point cloud research first began, new methods of point cloud generation were needed because point clouds were contained in irregular spaces with varying densities.

Initially, a method was proposed to convert 3D point cloud data into 2D images for processing. This method converts the points of 3D point cloud data into pixels of an image and treats them as images. While it has been successfully applied in the field of image

processing, it does not fully reflect the complexity and diversity of 3D point cloud data. Thus, other metrics are needed to process 3D point cloud data.

Initially, projection-based [4,5] and volumetric convolution-based methods [6–8] were proposed to convert each point into a grid to handle a 3D point cloud and perform feature extraction using convolutional layers in the same way as conventional 2D images on the grid. Because these methods convert irregular points in 3D space into a grid format, the number of points in the grid cell is uneven, resulting in the loss of information or wasted calculations in certain cells. To overcome the problems of grid transformation, direct point-based strategies have emerged. Some methods independently model each point using multiple shared multi-layer perceptrons (MLPs) [9–11]. Depending on the type of convolution kernel, 3D convolution methods have emerged [12–16].

Point clouds are being applied to various fields, such as classification, semantic segmentation [17,18], and registration [19], instance segmentation [20,21]. These methods use point cloud data as input and aggregate local features in the last step. While they maintain accurate location information, computational costs increase linearly with the number of points, and processing high-capacity, dense 3D point cloud data remains challenging. Accordingly, a method of sampling data is proposed to reduce the amount of data in the 3D point cloud and improve processing efficiency. Previously, heuristic-based sampling methods, such as task-agnostic random sampling, fast point sampling, and grid voxel sampling, have been used. However, these methods can degrade performance because they lose information or select meaningless data from downstream tasks. Recently, a task-oriented sampling network [22–24] was proposed, allowing the generation of sampling that optimized the performance of downstream tasks. S-Net and SampleNet performed well for specific tasks with sampling strategies using deep learning. In addition, APSNet used the attention-based method to focus on relationships among the points. Still, these models do not fully consider the relationship information between point clouds.

In this paper, we propose a methodology that leverages the complete information from the input sequence to effectively interact with the task model for task-oriented sampling. TransNet is a novel transformer-based model that handles an entire sequence in parallel, capturing a long range of point cloud information and point-to-point interaction information more effectively. Feature extraction is performed by adding the embedding layer of the input and positional encoding. After generating the query, key, and value, it proceeds through the transformer [25] layer with the self-attention mechanism to effectively capture the complex interrelationships among points within the input point cloud data. By focusing on the most informative points, our method can selectively sample only the most relevant areas of the point cloud, thereby improving the efficiency of the network and the accuracy of the output. Through this approach, we can gain a more comprehensive understanding of point cloud data and easily extract meaningful features that are essential for downstream tasks (see Figure 1). Our proposed model has achieved state-of-the-art performance improvements in the field of point cloud classification. In particular, the effect is remarkable for sparse points due to the high sampling ratio. To summarize, our main contributions are threefold:

1. We propose TransNet, a novel self-attention-based point cloud sampling network, as a task-oriented objective.
2. Our approach demonstrates enhanced performance on point cloud tasks, outperforming both task-agnostic and task-oriented methods.
3. This approach effectively addresses the long-range dependency issues that are commonly encountered in point clouds. Thus, it has a notable impact on the sparsely sampled point clouds, where a high sampling ratio is required to effectively capture the underlying geometric structures.

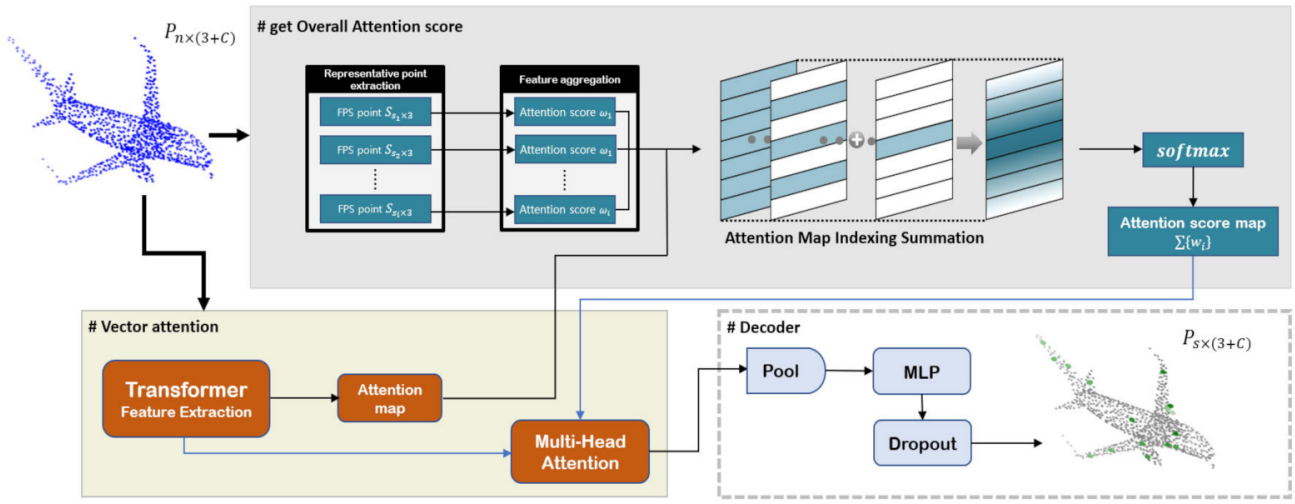

Figure 1. Overview of TransNet. TransNet divides the initial input into two parts. It calculates a comprehensive attention score map for each part (**top**) and vector attention (**lower left**). Then, multi-head attention is performed to understand the relationships between the points, and this process is repeated for training. In the decoder part (**lower right**), pooling and MLP are used to extract the final features. Detailed implementation methods are provided in Sections 3.1 and 3.2.

2. Related Work

Deep learning on point clouds: Deep learning has been applied to various point cloud-related tasks, such as object detection, segmentation, and classification. For example, Qi et al. [9,10] proposed point cloud classification and segmentation and achieved state-of-the-art performance on several benchmark datasets. In addition, a deep learning model was proposed for object detection in point clouds [26,27]. Other models have been proposed for point cloud processing utilizing local features [16] and adaptive convolution operations [28]. Additionally, some studies have explored the use of generative models [29,30] for point cloud generation and reconstruction tasks. One study [31] used a local self-attention mechanism, unlike the global attention scheme used in previous studies. Furthermore, it demonstrated that vector attention methods outperformed scalar attention methods and introduced position encoding methods to properly process location information in point clouds. Although the application was different in this paper, the self-attention technique was encoded by applying the point cloud technique similar to the point transformer [31]. To preserve the location information, positional encoding was utilized, and a decoder was constructed without undergoing multiple stages.

Point cloud sampling: Task-agnostic algorithms, such as random sampling, uniform sampling, farthest point sampling (FPS), and grid sampling, have been widely used in the past. Among them, FPS remains a popular choice in many recent studies [31,32]. While FPS has been widely used, it may not fully consider the downstream tasks for which the sampled points are used, leading to potential performance degradation. Thus, alternative downsampling methods have recently been proposed [22–24]. According to Dovrat et al. [22], the efficiency and accuracy of sampling could be improved through a learnable point cloud sampling method. Lang et al. [23] introduced a novel differentiable relaxation for point cloud sampling. The authors of [24] proposed sampling attention mechanisms to enhance the relationships among points by assigning importance weights, allowing for a more effective sampling process. Our TransNet is a task-oriented sampling method that mitigates long-range dependency while viewing the relationships between points globally and locally.

Transformer and self-attention: Transformer and self-attention models have revolutionized machine translation and NLP [25,33]. Considering this, such methods have been increasingly used in the field of 2D image recognition [34,35]. Inspired by these findings,

researchers have also attempted to apply self-attention networks to point cloud data. However, previous studies have utilized global attention on the entire point cloud, which limits their applicability to understanding large-scale 3D scenes due to high computational costs. Recently, Hengshuang et al. [31] developed a highly accurate and scalable self-attention network, specifically for large 3D scenes, using vector attention applied locally. In contrast to prior approaches, we applied a transformer locally to handle the input's point cloud sampling, which has been shown to be highly effective.

Nearest neighbor selection: In recent studies, nearest neighbor (NN) methods have been widely used for information fusion. However, in the context of neural networks, the main drawback is that the selection rule is not differentiable. To address this, Goldberger et al., proposed the probabilistic relaxation of NN rules by defining categorical distributions over a set of candidate neighbors. In our study, we applied self-attention to point clouds using k NNs, allowing for a better grasp of the relationships between the points. Additionally, we incorporated skip connections to consider information from the global area.

Positional encoding: In the domain of deep learning models, positional encoding has been commonly utilized to encode the positional information of input data. With respect to point clouds, previous research has employed basic encoding techniques, such as Cartesian, spherical, and polar coordinates, to incorporate the position information of the points. However, these methods have limitations in terms of information loss and insufficient expressiveness. To address this issue, some studies have employed learned positional encoding techniques to incorporate more informative position information into the model. These techniques usually involve learning a continuous function to represent the position information, which can capture complex spatial relationships and patterns in point clouds.

3. Proposed TransNet

Here, we briefly explain the transformer and self-attention concepts. Transformer and self-attention networks are innovative and have shown impressive results in NLP [25,33] and 2D image analysis [34,35]. Recently, networks have also been applied to 3D point cloud scenes [31,32]. Self-attention can be classified into two types: dot-product attention [25] and vector attention [36]. The standard formula for dot-product attention is as follows:

$$y_i = \sum_{x_j \in X} \tau\Big(\alpha(x_i)^T \beta(x_j) + \delta\Big)\gamma(x_j) \tag{1}$$

where $x_i \in X$ is a set of feature vectors, x_i and y_i are the input feature and output feature, respectively, α, β, and γ are pointwise feature transformations (e.g., MLP, linear layer), and τ and δ are normalization functions (a $softmax$ and a positional encoding function, respectively).

Unlike dot product attention, vector attention measures show similarity by calculating the distance between the input vector and the weight vector:

$$y_i = \sum_{x_j \in X} \tau\big(\varepsilon\big(\mu\big(\alpha(x_i), \beta(x_j)\big)\big) + \delta\big) \odot \gamma(x_j) \tag{2}$$

where μ is a relation function (e.g., subtraction, multiplication) and ε is a mapping function (e.g., MLP) that produces attention vectors for feature aggregation.

3.1. Transformer-Based Sampling Layer

Traditional task-oriented sampling methods, such as S-Net [22], SampleNet [23], and APSNet [24], use PointNet models that employ convolution networks to perform feature extraction. Moreover, S-Net and SampleNet generate m points at a time, and APSNet proceeds through the sequential generation process. In this study, we introduce a novel deep-learning sampling model based on self-attention. We processed inputs by defining the query, key, and value without using the convolution network. We used vector attention

and the subtraction operation between the query and key. Our vector attention process was as follows:

$$y_i = \sum_{x_j \in X_{knn}} \tau\left(\varepsilon\left(\mu\left(\alpha(x_i), \beta(x_j)\right)\right) + \delta\right) \odot \gamma(x_j) \quad (3)$$

Moreover, taking inspiration from [31], we performed self-attention within a local neighborhood to avoid the high computational costs that arise from global self-attention.

Here, $P \in R^{n \times 3}$ denotes a point cloud that contains a given point cloud n, which is the number of point clouds. We applied feature transformation to the local region selected by KNN to generate value v. We created an attention map of the same size as the value and used the indexing sum to create an attention score map W based on the relationship between representative points (a detailed explanation is given in Section 3.2). Finally, we obtained the attention value by using multi-head attention on the attention score map W and value:

$$Attention\ value = \sum_{i=1}^{n} w_i \odot v \quad (4)$$

3.2. Attention Score Map

We defined FPS as an algorithm for extracting representative points and performed multiple rounds of self-attention on the points extracted by FPS. With this result, we obtained the query and key for each sampled point, proceeded through a subtraction relationship, and then created a similarity in addition to Positional encoding. A description of the figure is shown in Figure 2. This process was repeated for the number of farthest point sampling performed. This resulted in the generation of an attention map that effectively encompassed all the generated values called the scatter sum. We describe the process in detail below.

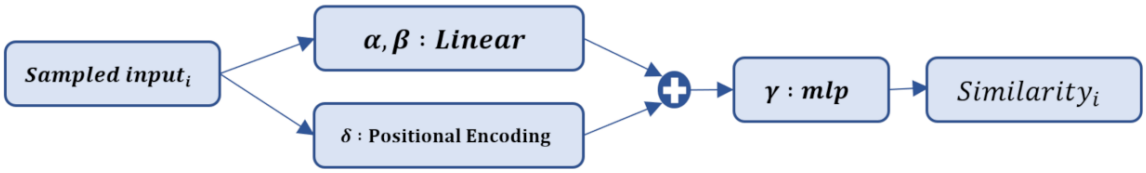

Figure 2. Similarity calculation. Prior to performing value v and multi-head attention, we generated attention scores for the sampled points via FPS and created an attention score map for multiple similarities. Additionally, we incorporated positional encoding to retain positional information.

First, we performed the KNN algorithm to find local features for the initial input and defined the value for local vectors. Then, we created an attention map for the empty space corresponding to the shape of the vector to calculate the distance between the input vector and the weight vector using vector attention. For the points S obtained through the FPS algorithm as $S \in P$, we generated query and key vectors for the representative S points among the N points generated through the FPS algorithm and examined their similarity through the subtraction relationship of the two generated vectors, as described in Equation (5).

$$w_i = \sum_{x_j \in X_{knn}} \alpha(x_i) - \beta(x_j) \quad (5)$$

We added this to the index corresponding to the attention score map W. We repeated this process for all set ratios R and applied a normalization activation function to the resulting attention score map:

$$W = idx(w_1) + idx(w_2) + \ldots + idx(w_i) \quad (6)$$

followed by performing multi-head attention with the initially obtained value v. Details are described in Figure 3.

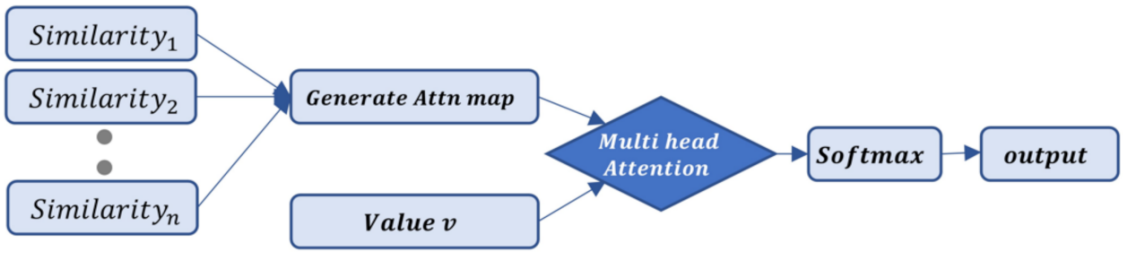

Figure 3. Transformer sampling layer.

To summarize the process before entering the Decoder, the initial point cloud was divided into two steps. In the first step (yellow area in Figure 1), the value v of the input was obtained, and an empty attention map of the same size was created. In the second step (gray area in Figure 1), a representative point was drawn through the FPS, and the query and key were obtained for each of the selected $S_{s_i \times 3}$; the similarity was obtained through this. Subsequently, we added weights to each index of the attention map generated in the first step to create a single attention map, which was then used to perform vector attention with the value v.

By identifying the interrelationships between the points in the input sequence, self-attention facilitated a better understanding of the relationships between each point, resulting in superior performance in handling long-term dependencies. Compared to traditional models that use LSTM [37], our proposed model enables the parallel processing of input sequences, resulting in superior performance, particularly in scenarios with high sampling ratios.

3.3. Decoder

In the decoding stage, a max-pooling operation was conducted to collect the extracted features. To generate the final output, a fully connected network (FCN) was utilized. To mitigate the loss of positional information that was caused by employing a linear MLP and positional encoding was further incorporated. Additionally, dropout was employed as a regularization technique to prevent overfitting. The result contained a number of s points for the task. With this result, we performed multi-task learning. By adopting a multi-task learning approach, all tasks could be efficiently processed within a single model. The concurrent training of these two models enabled us to effectively leverage the shared data distribution and consider more inter-task correlations, thus further improving the model's performance. Further details regarding the model's architecture are illustrated in Figure 1 (lower right).

3.4. Loss

We applied supervised learning and used two types of loss: task loss L_{task} and sampling loss $L_{sample,}$ to train TransNet, where Total loss is the sum of the weights added to these two losses. The sampling loss L_{sample} aimed to minimize the distance between the points sampled from S, and the corresponding points in P, while also ensuring that the sampled points were spread out as much as possible across the original point cloud P.

L_{task} can be defined as the cross-entropy loss for classification. Additionally, the formula for this is as follows:

$$L(\hat{y}, y) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \tag{7}$$

L_{sample} is the sum of two things: average neighbor loss and maximum neighbor loss. Given two-point sets Q_1 and Q_2, the average nearest neighbor loss can be denoted as:

$$L_a(Q_1, Q_2) = \frac{1}{|Q_1|} \sum_{q_1 \in Q_1} \min_{q_2 \in Q_2} ||q_1 - q_2||_2^2 \tag{8}$$

Additionally, maximal nearest neighbor loss can be given as:

$$L_m(Q_1, Q_2) = \max_{q_1 \in Q_1} \min_{q_2 \in Q_2} ||q_1 - q_2||_2 \tag{9}$$

The sampling loss is then given by:

$$L_{sample}(Q, P) = L_a(Q, P) + \beta L_m(Q, P) + (\gamma + \delta|Q|)L_a(P, Q) \tag{10}$$

where β, γ and δ are the hyperparameter that adjusts the size between losses. In conclusion, the total loss is then as follows:

$$L_{total} = L_{task} + \lambda L_{sample}(Q, P) \tag{11}$$

where λ is a hyperparameter value that adjusts the value between the task loss and the sample loss.

4. Experimental Results and Discussion

In this section, we demonstrate that the performance of our TransNet is superior to that of existing sampling methods in various fields. We conducted experiments in the classification and registration domains and proved that the performance was particularly good in areas with high point cloud sampling ratios. We experimented with two variations of TransNet (W.O indexing summation) and TransNet. The former is a model that applies the transformer architecture. This is a method of applying the Transformer method by generating queries, keys, and values based on existing information without using FPS points. The latter is a model that incorporates an attention map into the transformer architecture. This last model performed better, and here we experimented by comparing the latter model with those of other papers.

We implemented TransNet in PyTorch, setting the batch size, SGD optimizer with momentum, and weight decay to 128, 0.9, and 0.0001, respectively. In addition, we conducted 400 epochs of training in all the experiments. For classification, we used the ModelNet 40 dataset [38]. We performed experiments on 1024 points that were uniformly sampled. To train and evaluate our models, we used the train-test split dataset provided on the official website. We used instance-wise accuracy as a metric to evaluate the classification results of each sample in multi-class classification problems. Each sample belonged to a single class, and if the predicted class by the classification model matched the actual class, the sample was considered "correctly classified". Therefore, instance-wise accuracy represents the proportion of samples that were correctly classified among all the samples. Furthermore, we focused our experiments on sparse points with sampling ratios of 16, 32, 64, and 128, which had previously shown an inferior performance in all papers. In this study, we conducted an experimental analysis using PointNet, which led us to assume the outcomes of Table 1.

The sampled experiment would not surpass the accuracy achieved by the original PointNet prior to sampling. Therefore, we contended that using other state-of-the-art models can also lead to higher sampling accuracy. For example, in the case of classification, since PointNet was defined as the underlying model, it was assumed that the performance of the unsampled PointNet would not be exceeded by any sampled model.

As Figure 4 shows, we have created a model that exceeds the performance of existing models for the experimental results of sampling 8, 16, 32, 64 for 1024 points each. In particular, we demonstrated that the sparse points (the results of sampling 8 or 16) resulted

in greater deviations from other models and that our model was more robust in sparse data. In Figure 5, we present a sampling comparison experiment of our model and comparison model on the same object, from which we can clearly see the difference. Our model samples objected much more evenly and reasonably than an APSNet model's sampling result, demonstrating its superior performance in classification results. As a result, our TransNet had a better grasp of corners and characteristic parts than existing methods. However, all deep learning models exhibited a tendency to mislead in certain areas (table legs, flower in a pot, etc.), with many weak characteristics in common, and this problem remains to be solved.

Table 1. Classification accuracies of five sampling methods on ModelNet40. All experiments were conducted in the same environment.

Sampling Ratio	128	64	32	16
RS	8.7	24.87	54.53	79.26
FPS	24.3l	55.12	76.92	87.53
SampleNet [23]	80.71	85.32	86.38	87.10
APSNet [24]	82.72	84.89	86.66	**88.00**
TransNet	**87.47**	**88.16**	**88.49**	87.88

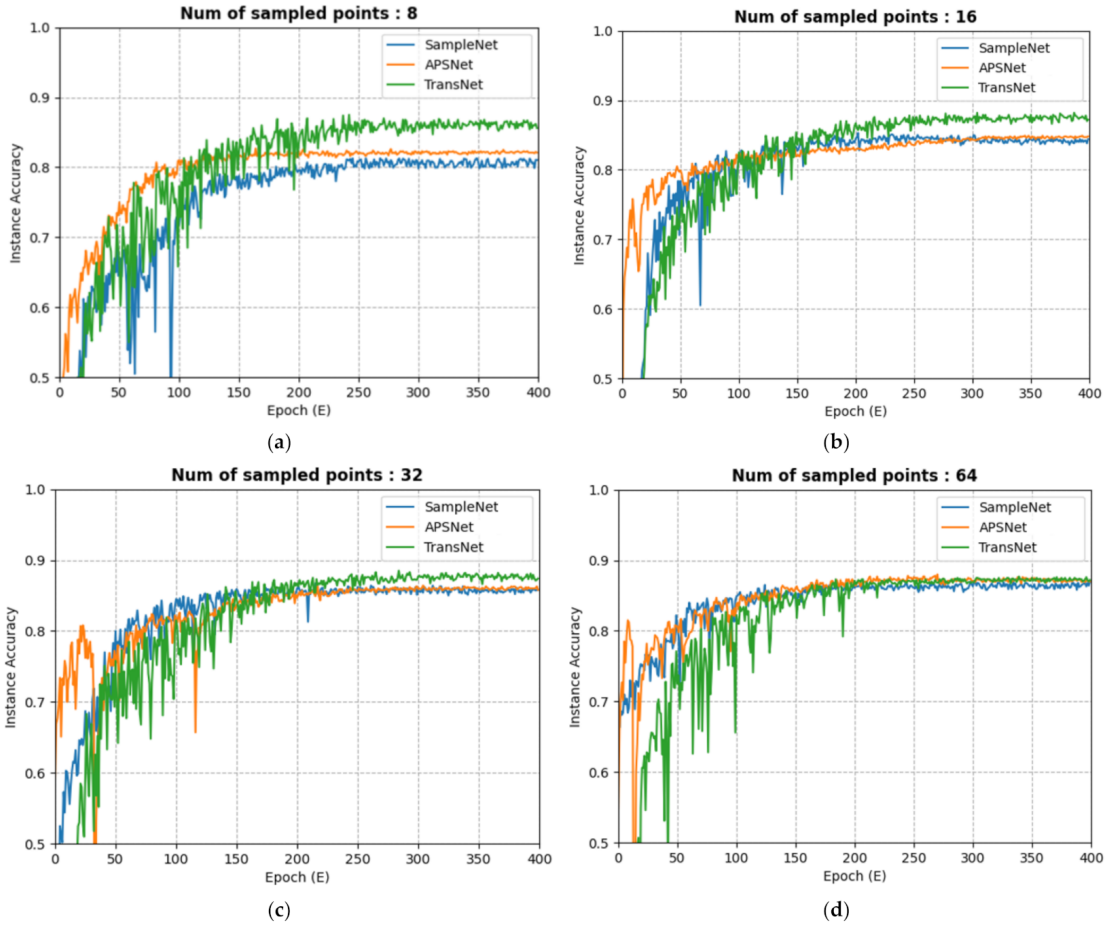

Figure 4. Instance accuracy. This is the result of training three sampling models (SampleNet, APSNet, TransNet) on the ModelNet40 dataset after uniformly sampling 1024 examples. The sampling ratios used were 128, 64, 32, and 16. For example, if 8 examples were sampled from 1024, the sampling ratio would be 128. TransNet achieved much better performance in sparse areas such as (**a**,**b**) and showed superior performance in other areas. Less sparse (**c**,**d**) can also see similar or higher levels of results than existing papers. The classification accuracies are shown in Table 1.

Figure 5. Visualized Sampled Points. Results for 8, 16, and 32 sampling points (green) on ModelNet40. The results on the left are from TransNet (**a**), and those on the right are from APSNet (**b**). Moreover, from top to bottom, the objects are a laptop, nightstand, chair, and airplane. The gray points represent the original ground truth, and the green points were generated. From the overall shape, the result value of our TransNet (**a**) szx more evenly expressed than the result value of APSNet (**b**). Overall, for square objects (laptop, desk), the method tended to have a better grasp of the ends, while for objects such as chairs, it tended to have a better grasp of the legs. Detailed class-specific results are shown in Table 2.

Table 2. Class-specific accuracies of two sampling methods on ModelNet40. All experiments were conducted in the same environment.

Sampling Ratio		Laptop	Chair	Nightstand	Airplane
128	SampleNet	81.32	91.38	38.18	97.51
	APSNet	83.33	95.56	42.18	98.52
	TransNet	**85.32**	**96.49**	**51.88**	**99.1**
64	SampleNet	85.23	94.38	59.38	98.11
	APSNet	88.88	95.52	64.67	99.49
	TransNet	**91.42**	**96.52**	**65.92**	**99.55**
32	SampleNet	89.29	86.38	67.21	98.89
	APSNet	90.90	94.68	67.39	**99.50**
	TransNet	**90.91**	**97.05**	**72.50**	99.00

5. Conclusions

Transformer algorithms have been expanded beyond the natural language process into various fields. We applied this algorithm to the field of point cloud sampling and achieved successful results. In this paper, we proposed a novel transformer-based point cloud sampling network to achieve precision performance. While S-Net and SampleNet generated the sampling process using MLP-based methods, and APSNet used an attention-based model, TransNet employed a multi-head self-attention technique in the down-sampling process. In addition, assuming that FPS is a representative point for viewing general-purpose information, we created an attention map that collected information after proceeding through several FPS algorithms and indexed it for each location while applying multi-head attention. This strategy improved the relationship between each point in the learning procedure while it also learned simultaneously with task models and reasonably understood relationships alongside alleviating long-range dependencies. The proposed TransNet demonstrated a better performance in terms of precision, especially on sparse data. Moreover, the proposed sampling method could be applied to various kinds of point cloud deep learning networks. Thus, its usefulness would be meaningful in many practical scenarios.

6. Ablation Study

6.1. K-Nearest

After extracting the representative points with FPS, we applied the K-nearest neighbor algorithm to find the neighbor points for the representative points and identify the association between the points. The neighborhood size k is the number of neighbors in the representative point P of a point $q \in Q$. We evaluated the impact of the hyperparameter, k, by training multiple progressive TransNet for classification with different values of k. TransNet was applied as k = 16, and experiments were conducted on $k \in \{4, 8, 16, 32\}$, respectively. The results of the experiment are shown in Table 3. If the neighbor was smaller (k = 4 or k = 8), there might not have been enough context for the model to make predictions. If the neighbor was larger (k = 32), each self-attention layer had a large number of data points, many of which could be further away and less relevant. This could result in excessive noise during processing with the potential to degrade the accuracy of the model.

Table 3. The accuracy of TransNet according to K. All experiments were conducted in the same environment, and we adopted K = 16.

K-Size	TransNet-4			TransNet-8			TransNet-16			TransNet-32		
Sampling ratio	128	64	32	128	64	32	128	64	32	128	64	32
Instance Accuracy	85.36	86.24	85.32	85.95	86.68	85.93	**87.47**	**88.16**	**88.49**	86.21	87.67	87.58

6.2. Additional Experiments

As mentioned briefly earlier, when applying the Task model without the sampling model, higher accuracy was achieved compared to the application of the sampling model. When tested with PointNet, an accuracy of 90.08 was obtained. To be as close to this target as possible, we made several attempts, such as adding dropout, skipping connections, or experimenting with added positional encoding to add positional information in different parts of the model. The activation function also conducted many experiments, such as ReLu, Leaky ReLu, and ELU. As a result, we set the probability of the dropout to 0.1 and added positional encoding to the encoder and decoder portions of the model, respectively. The activation function performed best when using ReLu.

Author Contributions: Conceptualization and formal analysis, H.L.; investigation and validation, J.J.; methodology and software, S.H.; software and writing, J.K. and J.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) grants funded by the Korean government (MSIT) (NRF-2021R1A5A1032937).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liang, Z.; Guo, Y.; Feng, Y.; Chen, W.; Qiao, L.; Zhou, L.; Zhang, J.; Liu, H. Stereo matching using multi-level cost volume and multi-scale feature constancy. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 300–315. [CrossRef] [PubMed]
2. Guo, Y.; Sohel, F.; Bennamoun, M.; Lu, M.; Wan, J. Rotational projection statistics for 3D local surface description and object recognition. *Int. J. Comput. Vis.* **2013**, *105*, 63–86. [CrossRef]
3. Guo, Y.; Bennamoun, M.; Sohel, F.; Lu, M.; Wan, J. 3D object recognition in cluttered scenes with local surface features: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 2270–2287. [CrossRef] [PubMed]
4. Lawin, F.J.; Danelljan, M.; Tosteberg, P.; Bhat, G.; Khan, F.S.; Felsberg, M. Deep Projective 3D Semantic Segmentation. In Proceedings of the International Conference on Computer Analysis of Images and Patterns, Ystad, Sweden, 22–24 August 2017; pp. 95–107.
5. Boulch, A.; Le Saux, B.; Audebert, N. Unstructured Point Cloud Semantic Labeling Using Deep Segmentation Networks. In Proceedings of the Workshop 3D Object Retrieval, Lyon, France, 23–24 April 2017; pp. 17–24.
6. Maturana, D.; Scherer, S. Voxnet: A 3d Convolutional Neural Network for Real-Time Object Recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–2 October 2015.
7. Riegler, G.; Ulusoy, A.O.; Geiger, A. Octnet: Learning Deep 3d Representations at High Resolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
8. Wang, P.-S.; Liu, Y.; Guo, Y.X.; Sun, C.Y.; Tong, X. O-cnn: Octree-based convolutional neural networks for 3d shape analysis. *ACM Trans. Graph. (TOG)* **2017**, *36*, 1–11. [CrossRef]
9. 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, Honolulu, HI, USA, 21–26 July 2017.
10. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In Proceedings of the 2017 Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Volume 30.
11. Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R.R.; Smola, A.J. Deep sets. In Proceedings of the 2017 Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Volume 30.
12. Song, W.; Liu, Z.; Tian, Y.; Fong, S. Pointwise CNN for 3d object classification on point cloud. *J. Inf. Process. Syst.* **2021**, *17*, 787–800.
13. Thomas, N.; Smidt, T.; Kearnes, S.; Yang, L.; Li, L.; Kohlhoff, K.; Riley, P. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv* **2018**, arXiv:1802.08219.
14. Groh, F.; Wieschollek, P.; Hendrik; Lensch, P.A. Flex-Convolution: Million-Scale Point-Cloud Learning Beyond Grid-Worlds. In *Proceedings of the Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, 2–6 December 2018*; Revised Selected Papers, Part I 14; Springer International Publishing: Berlin/Heidelberg, Germany, 2019.
15. Wu, W.; Qi, Z.; Fuxin, L. Pointconv: Deep Convolutional Networks on 3d Point Clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.

16. Thomas, H.; Qi, C.R.; Deschaud, J.E.; Marcotegui, B.; Goulette, F.; Guibas, L.J. Kpconv: Flexible and Deformable Convolution for Point Clouds. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019.
17. Li, J.; Chen, B.M.; Lee, G.H. So-net: Self-Organizing Network for Point Cloud Analysis. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018.
18. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic graph cnn for learning on point clouds. *ACM Trans. Graph. (TOG)* **2019**, *38*, 1–12. [CrossRef]
19. Aoki, Y.; Goforth, H.; Srivatsan, R.A.; Lucey, S. Pointnetlk: Robust & Efficient Point Cloud Registration Using Pointnet. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
20. Wang, W.; Yu, R.; Huang, Q.; Neumann, U. SGPN: Similarity Group Proposal Network for 3d Point Cloud Instance Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–22 June 2018.
21. Wang, X.; Liu, S.; Shen, X.; Shen, C.; Jia, J. Associatively Segmenting Instances and Semantics in Point Clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
22. Dovrat, O.; Lang, I.; Avidan, S. Learning to Sample. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
23. Lang, I.; Manor, A.; Avidan, S. SampleNet: Differentiable Point Cloud Sampling. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020.
24. Ye, Y.; Yang, X.; Ji, S. APSNet: Attention Based Point Cloud Sampling. *arXiv* **2022**, arXiv:2210.05638.
25. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In Proceedings of the 2017 Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017; Volume 30.
26. Shi, S.; Wang, X.; Li, H. Pointrcnn: 3D Object Proposal Generation and Detection from Point Cloud. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019.
27. Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X.; Li, H. Pv-rcnn: Point-Voxel Feature Set Abstraction for 3d Object Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020.
28. Jiang, M.; Wu, Y.; Zhao, T.; Zhao, Z.; Lu, C. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. *arXiv* **2018**, arXiv:1807.00652.
29. Yang, G.; Huang, X.; Hao, Z.; Liu, M.Y.; Belongie, S.; Hariharan, B. Pointflow: 3d Point Cloud Generation with Continuous Normalizing Flows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019.
30. Vakalopoulou, M.; Chassagnon, G.; Bus, N.; Marini, R.; Zacharaki, E.I.; Revel, M.P.; Paragios, N. Atlasnet: Multi-Atlas Non-Linear Deep Networks for Medical Image Segmentation. In *Medical Image Computing and Computer Assisted Intervention–MICCAI 2018: 21st International Conference, Granada, Spain, September 16–20, 2018, Part IV 11*; Springer International Publishing: Berlin/Heidelberg, Germany, 2018.
31. Zhao, H.; Jiang, L.; Jia, J.; Torr, P.H.; Koltun, V. Point Transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021.
32. Lai, X.; Liu, J.; Jiang, L.; Wang, L.; Zhao, H.; Liu, S.; Qi, X.; Jia, J. Stratified Transformer for 3d Point Cloud Segmentation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022.
33. Wu, F.; Fan, A.; Baevski, A.; Dauphin, Y.N.; Auli, M. Pay less attention with lightweight and dynamic convolutions. *arXiv* **2019**, arXiv:1901.10430.
34. Hu, H.; Zhang, Z.; Xie, Z.; Lin, S. Local Relation Networks for Image Recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019.
35. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021.
36. Zhao, H.; Jia, J.; Koltun, V. Exploring Self-Attention for Image Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020.
37. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
38. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3d Shapenets: A Deep Representation for Volumetric Shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015.