IEEE Access
Multidisciplinary Rapid Review Open Access Journal

Received 11 August 2023, accepted 28 September 2023, date of publication 10 October 2023, date of current version 27 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3323428

RESEARCH ARTICLE

PMHA-Net: Positional Multi-Head Attention Network for Point-Cloud Part Segmentation and Classification

JAESEUNG JEON[1], SEOKJIN HONG[1], HOOKYUNG LEE[1], JEESU KIM[2], (Member, IEEE), AND JINWOO YOO[3], (Member, IEEE)

[1]Graduate School of Automotive Engineering, Kookmin University, Seoul 02707, Republic of Korea
[2]Departments of Cogno-Mechatronics Engineering and Optics & Mechatronics Engineering, Pusan National University, Busan 46241, Republic of Korea
[3]Department of Automobile and IT Convergence, Kookmin University, Seoul 02707, Republic of Korea

Corresponding author: Jinwoo Yoo (jwyoo@kookmin.ac.kr)

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

ABSTRACT For understanding unordered sets of point clouds, the positional information of each point must be effectively used. To this end, the existing models use the absolute position of the point and its relative position within a local group. However, accurately capturing the positional information is challenging because the relative position within a local group typically has a considerably smaller value than that of the feature information. Moreover, in terms of the data characteristics of point clouds, closer points are more strongly correlated, but their relative position approaches zero. To address these problems, we process the relative position within a local group by normalizing it within the overall object range and local range according to the data characteristics. This transformation helps maintain the meaning and pattern of the relative position while facilitating its learning. The transformed data are combined with the absolute position to encode the position vector, which serves as the positional encoding in multi-head attention across multiple resolutions. Extensive experiments are conducted on benchmark point cloud datasets to demonstrate that the proposed model exhibits competitive performance in part segmentation and classification tasks.

INDEX TERMS Point cloud, deep learning, multi-head attention, positional encoding, up-sampling, part segmentation, classification.

I. INTRODUCTION

A point cloud, typically derived from LiDAR or three-dimensional (3D) camera scans, is a representative data-driven tool for depicting the environment in three dimensions. Point clouds consist of a large number of 3D coordinate points, which contain important spatial information, such as the shape, location, and size of an object. Consequently, point clouds are widely used in various fields, such as robotics [1], computer vision [2], autonomous driving [3], and virtual reality. With recent advancements in deep learning and 3D scanning sensors, the affordability of 3D environment recognition has improved. Moreover, the development of improved hardware for parallel computation processing has promoted research on point clouds. The objective of point-cloud deep learning is to maximize the extraction of accurate and high-dimensional characteristics by applying deep learning algorithms to 3D data.

A. POINT-CLOUD DEEP LEARNING TASK

1) CLASSIFICATION

As a fundamental deep learning task, classification involves predicting the category to which a particular object or scene belongs. The objective is to classify the input point cloud to a predefined category. To this end, a model is trained to correctly categorize input point clouds into labels such as airplanes, cars, or desks.

The associate editor coordinating the review of this manuscript and approving it for publication was Davide Patti.

© 2023 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

2) SEGMENTATION

The segmentation of a point cloud is different from classification in that every point has a label. Subtasks of segmentation include semantic segmentation and part segmentation. Semantic segmentation is the task of classifying which object each point is contained in a point cloud scene with multiple objects. Part segmentation is aimed at predicting the part to which each point of an object belongs. For example, the point cloud of an airplane is divided into parts such as wings, bodies, and wheels. This task enhances the ability of a model to understand the object structure and perceive the detailed shape and characteristics.

3) REGISTRATION

Point cloud registration involves aligning different point clouds into a coherent 3D model. For example, SpinNet [4] extracts rotationally invariant and informative local features to enhance registration accuracy and has been noted to outperform the existing methods on various datasets.

4) SAMPLING

This task involves selecting representative points from the point cloud to achieve a satisfactory performance on the aforementioned tasks, even with a small number of points. For example, TransNet [5] uniformly samples the point cloud at various scales using the farthest point sampling strategy. By obtaining multi-scale attention weights, TransNet [5] achieves satisfactory classification performance even with a few points.

The abovementioned tasks allow deep learning models to effectively extract and understand spatial and semantic information from point cloud data. The objective of this study is to understand the basic properties of point clouds and propose more effective algorithms for point cloud classification and segmentation.

B. OVERVIEW OF POINT-CLOUD DEEP LEARNING

Point clouds consist of 3D points, each characterized by x, y, and z coordinates. Point-cloud deep learning typically involves tasks such as classification and segmentation, using both global and local features. Global features contain overall shape information, whereas local feature contain information regarding the local shape of an object. Notably, the performance of refined classification or detection of subtle shape changes may deteriorate if only global features are used. In such case, it is necessary to integrate local features. To extract local features, information regarding the points surrounding each point must be obtained. This requires a grouping process, which identifies the indices of the points around a given point. Grouping is generally based on the Euclidean distance along with techniques such as the k-nearest neighbor (kNN) algorithm, which selects the k nearest points, or the ball query method, which selects all points within a given radius. To characterize local features, it is necessary to determine not only the absolute position of the points but also the relational and relative positions of grouped points. Each point distributed in 3D space holds limited information in terms of the absolute position. However, by considering the arrangement of points, that is, the relative positional relationship between the points, more complex and detailed shape information can be extracted. For instance, information such as the distance and orientation of a point relative to other points or the distribution of distances to surrounding points can provide valuable clues regarding the shape, texture, and structure of the object that the point belongs to. In the Point Transformer framework [6], the relative position in each 3D axis is directly used in learning by incorporating the positional information in the feature vector.

However, the direct use of the relative position involves certain challenges. First, the relative position values are small. As point cloud data become denser, the relative positional values between two points become extremely small. Ideally, the positional information value should not be excessively large or small. Specifically, an extremely small value may cause the positional information to have no meaning in the learning process. Conversely, if the positional information is excessively large, the influence of other encoded information may be reduced. Therefore, it is necessary to ensure the appropriate scale of the positional information. Second, the magnitude of the relative position decreases as two points draw closer. Two points with a small distance between them are likely to have similar characteristics, and thus, they are strongly correlated. However, the relative position approaches zero as the distance between the two points decreases. In such cases, even though the two points are correlated, the positional information entering the learning layer approaches a value of zero. If the value of a specific index in the input data approaches zero, the model tends to pay less attention to the corresponding point during training. This phenomenon occurs because most learning algorithms learn weights associated with the patterns in the input data, and these weights are determined through the dot product operation with the input data. If the value at a specific index is close to zero, the corresponding weight does not significantly contribute to the result of the dot product operation with the input. Therefore, it is desirable to assign higher scores to two points with smaller distances. This adjustment allows the model to be more influenced by nearby points than by distant ones during training. By normalizing the relative position data to an appropriate scale while maintaining the pattern and meaning of the data, the positional information of the point cloud can be more effectively utilized.

Considering these aspects, in this study, we construct a multi-head attention mechanism with positional encoding for reflecting the point cloud distribution characteristics of an object. Moreover, a method for positional encoding that is suitable for point clouds is established. To this end, we obtain a relative position score between the points in the sampled point cloud and the points before being sampled. The relative positional information used in existing models is transformed according to the characteristics of each object while maintaining an appropriate scale, thereby allowing the learning model to leverage the positional information more effectively. The position score is designed to increase as the distance between the points decreases, and it is normalized within the appropriate scales.

To properly reflect the original patterns and characteristics of the data during normalization, the adjustment range must be carefully defined. In our analysis, each object in a dataset is noted to have different point densities and distributions. Therefore, object-specific normalization is performed in the proposed framework. Additionally, after observing that the density of the point cloud varies by part within a single object, we also concurrently implemented normalization within the local range. And considering the 3D characteristics of point clouds, we maintain a score for each axis and preserve the sign indicating the direction with respect to the center point.

The main contributions of this study can be summarized as follows:

- We introduce a positional encoding method that is suitable for point cloud data and can be applied to existing networks.
- A multi-head attention-based model is established, which can effectively use positional encoding to facilitate appropriate utilization of positional information in multi-head attention mechanisms.
- The proposed positional encoding approach is applied to up-sampling processes, whose development has been known to be stagnant.

The remaining paper is organized as follows. Section II provides a review of existing point-cloud deep learning methods and related literature. Section III outlines the proposed multi-head attention and positional encoding methods. Section IV discusses the experimental results. Section V presents the concluding remarks and highlights future research directions.

II. RELATED WORK

A. POINT-CLOUD DEEP LEARNING METHODS

Various methods have been developed for learning point clouds. For example, voxel-based approaches [7], [8], [9], [10] convert a point cloud into a 3D grid and perform shape classification for volumetric data using 3D convolutional neural networks (CNNs). VoxNet [7], a representative voxel-based method, uses 3D-CNNs to perform 3D object recognition. In general, two-dimensional (2D) CNNs can effectively encode spatial relationships between image pixels; 3D-CNNs extend this ability to 3D space to encode the spatial relationships of 3D data. Voxel-based methods played a key role in early 3D object recognition research. However, these methods tend to lose part of the detailed information and are thus inefficient in handling sparse point-cloud data.

Multi-view based methods represent another class of point-cloud-learning strategies [11], [12], [13], [14]. These methods convert complex spatial information from 3D objects into 2D image features. This involves converting 3D objects into 2D images viewed from various angles, extracting features from each of these "views", and combining them to classify 3D objects. For example, MV-CNN [11] renders 3D models into multiple 2D views and passes each 2D view through an independent CNN. The features obtained from each CNN are combined to obtain a comprehensive representation of the 3D model. Although multi-view based methods can handle point clouds, they exhibit several limitations: A scarcity of views may lead to loss of 3D information. Moreover, because each view is processed independently, the model may not effectively capture interrelationships. Increasing the number of views may be computationally expensive as it requires the generation of multiple views, feature extraction from each view, and their fusion.

In graph-based methods [15], [16], [17], [18], point clouds are presented in a graphical form and processed using graph convolutional networks. A notable example is DGCNN [15], which constructs a dynamic graph by finding the nearest neighbors for each point in the point cloud and then applies a convolution to the graph. Graph-based methods can effectively model the spatial structure of 3D point cloud data. However, models that use dynamic graphs must reconstruct the graph at each layer of the network, resulting in high computational and memory costs. These costs may increase prohibitively in the case of large-scale point clouds with a large number of points and deep networks.

Unlike the abovementioned methods, point-based methods [6], [19], [20], [21] directly use raw point clouds without any preprocessing steps. The absolute position of points in terms of x, y, and z coordinates is used as the input data. Depending on the model, RGB data or normal vectors may also be used. A fundamental problem with point clouds is that they represent unordered datasets. PointNet [19] addressed this problem by using a max pooling function to ensure permutation invariance, thereby yielding consistent results even when the order of the input points varies. Building upon PointNet [19], PointNet++ [20] mitigates information imbalance by incorporating local information in addition to global information through a hierarchical structure and skip connections. This hierarchical structure of PointNet++ [20] has been applied to many models that use raw point clouds [6], [21], [22], [23], [24].

B. MULTI-HEAD ATTENTION

Self-attention can effectively model interactions within input data and has demonstrated excellent performance in natural language processing and machine translation tasks [25]. Therefore, self-attention can be applied to point clouds to model relationships with neighboring points in 3D space. Because point clouds exist in 3D space and the data are unordered, identifying the relative position to neighboring points is crucial for understanding local information through self-attention.

Multi-head attention can be considered an extension of self-attention. This strategy divides the number of input

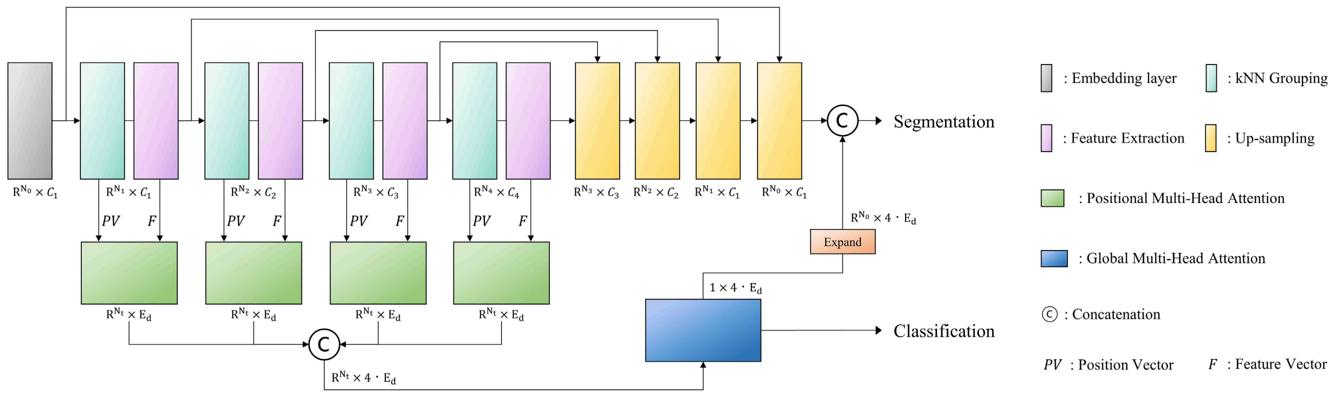

FIGURE 1. Overall architecture of PMHA-Net.

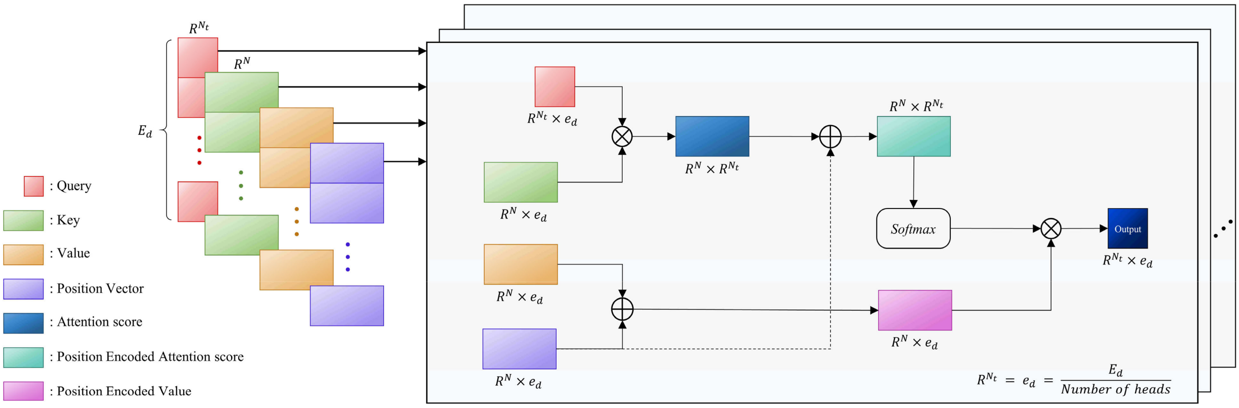

FIGURE 2. Process of positional multi-head attention. The outputs from each head are concatenated to form the final result.

feature channels by the number of heads, independently processes each split sequentially, and then combines them.

Because the divided input features independently perform self-attention calculations, they can learn patterns from different perspectives and ensure diversity in interpretation.

C. POSITIONAL ENCODING

Self-attention was originally developed for sequential data such as sentences. Therefore, to apply this mechanism to the 3D structural data of unordered point clouds, the positional information for each data point must be known. To achieve this, the positional encoding [6], [25], [26] mechanism must be introduced to recognize the spatial structure of point clouds. The location information includes the absolute position of each point and relative position of neighboring points. The Point Transformer [6] method allows a model to effectively learn the spatial structure of the point cloud by directly incorporating the relative positional information of each point into the self-attention mechanism. In point cloud data, the points defining the shape or structure of an object are typically located close to one another, and the relative positional relationship between these points is pivotal for identifying the object shape.

In this study, we establish the positional multi-head attention (PMHA) mechanism, which reinterprets relative positional information based on the characteristics of 3D point clouds. This approach uses both the absolute and relative positional information of point clouds, thereby effectively considering structural characteristics, and is expected to exhibit robust performance in point-cloud deep learning tasks.

III. PMHA-NET

A. POINT CLOUD

Point clouds can be obtained from 3D cameras or LiDAR scans. Point clouds in 3D space can be denoted as $S_N \in R^N \times 3$, with each of the N points constituting the point cloud represented as p_i:

$$S_N = \{p_1, p_2, \ldots, p_N\} \tag{1}$$

Point clouds of the same object may be arranged in diverse combinations depending on the order of the points. Thus, a given object may have $N!$ different representations. This variability due to the order of the data may serve as an unstable factor affecting the classification results of a neural network. Considering the unordered nature of point cloud data, we use the max pooling function to generate features that are independent of the order. This strategy is crucial to minimize the influence of changes in the order of point cloud data and yield consistent classification results.

B. NETWORK ARCHITECTURE

Fig. 1 shows the overall network of PMHA-Net for learning point clouds at multiple resolutions. The network consists of kNN grouping and feature extraction modules for extracting local information, PMHA for obtaining diverse perspectives on the relationships between points in each layer, and global multi-head attention (GMHA) for combining information across layers. The feature extraction module involves residual blocks incorporating batch normalization [27] and rectified linear unit (ReLU) activation functions, following the PointMLP framework [22]. Fig. 2 presents the architecture of PMHA, which embodies our proposed positional encoding concept for point-cloud deep learning. This method, unlike conventional multi-head attention, adds a redesigned position vector PV, which is divided by the number of heads for independent use and combined later. The proposed strategy facilitates the acquisition of the location information of points at multiple resolutions and interpretation of the location information from multiple perspectives through the attention from multiple heads.

1) POSITIONAL MULTI-HEAD ATTENTION

This section describes the use of the position vector $PV \in R^N \times E_d$ in the multi-head attention of the PMHA framework shown in Fig. 2. The input S_N of the network passes through the embedding layer and feature extraction module to produce the feature vector $F \in R^N \times C$. The feature vector F passes through two multi-layer perceptrons (MLPs) to generate the key $K \in R^N \times E_d$ and value $V \in R^N \times E_d$. The query $Q \in R^{N_t} \times E_d$ adopts a learnable query based on the learnable pooling of PointStack [21]. Through this learnable query, we can minimize the information loss caused by max pooling and use a query that is closely related to the learning objective. Q, K, V, and PV are unified to the embedding dimension $\mathrm{E_d}$ through the MLP. Q remains fixed at R^{N_t} regardless of the scale of the point cloud. As illustrated in Fig. 1, because down-sampling stages are included, the output of PMHA $Y_N = \{Y_1, Y_2, Y_3, Y_4\}$ is yielded by four different resolutions. The method for calculating the k-th output Y_k is described in the following text. Before they are input to the multi-head attention, the channel dimensions of Q, K, V, and PV are portioned by the number of heads h, as follows:

$$Q_N = \{q_1, q_2, \ldots, q_N\} \tag{2}$$

$$K_N = \{k_1, k_2, \ldots, k_N\} \tag{3}$$

$$V_N = \{v_1, v_2, \ldots, v_N\} \tag{4}$$

$$PV_N = \left\{pv_1, pv_2, \ldots, pv_N\right\} \tag{5}$$

Subsequently, we calculate the partial attention according to the order of each head, and the output y_i of the i-th head of PMHA is derived:

$$y_i = softmax(q_i \times k_i^T + pv_i) \times (v_i + pv_i) \tag{6}$$

Next, the h outputs calculated by splitting are concatenated to obtain the output $Y_k \in R^{N_t} \times E_d$ of multi-head attention, as follows:

$$Y_k = concat(y_1; y_2; \ldots; y_N) \tag{7}$$

Overall, PMHA unifies the channel dimensions of the query, key, value, and position vector to the embedding dimension E_d and sets the query as R^{N_t}, regardless of the layer. Therefore, all outputs from the four resolutions $Y_k \in R^{N_t} \times E_d$ have the same shape. And we aggregate these four outputs, denoted as $\{Y_N = Y_1, Y_2, Y_3, Y_4\}$.

2) GLOBAL MULTI-HEAD ATTENTION

Subsequently, all elements in Y_N are concatenated to obtain $Y \in R^{N_t} \times 4 \cdot E_d$. And we apply the Global Multi-Head Attention (GMHA) to get $Y_G \in 1 \times 4 \cdot E_d$ which is a combination of the information from multiple resolutions. GMHA performs the same operation as PMHA, except that it does not use the position vector, as follows:

$$y_{Gi} = softmax(q_{Gi} \times k_{Gi}^T) \times (v_{Gi}) \tag{8}$$

$$Y_G = concat(y_{G1}; y_{G2}; \ldots; y_{GN}) \tag{9}$$

where q_{Gi}, k_{Gi}, and v_{Gi} are the split query, key and value for GMHA, respectively; and y_{Gi} is the i-th head output of GMHA.

Y_G is used for classification or is expanded and concatenated with the results of up-sampling for segmentation. The classification and segmentation tasks involve a task-specific MLP, which includes batch normalization, ReLU, and dropout with a 0.5 drop rate. Classification results are obtained through a linear layer, and segmentation results for each point are extracted through a one-dimensional convolution layer.

3) POSITIONAL ENCODING

The application of the position vector PV as a positional encoding tool with the query, key, and value within a single head is inspired by the positional encoding method of the Point Transformer [6], which can be represented as:

$$\sum\nolimits_{\mathrm{x_j} \in X(i)} \rho(\gamma(\varphi\,(\mathrm{x_i}) - \psi(\mathrm{x_j}) + \delta)) \odot (\alpha(\mathrm{x_j}) + \delta) \tag{10}$$

where $X = \{x_i\}_i$ is a set of feature vectors; the subset $X\,(i) \subseteq \mathrm{X}$ is a kNN group; and φ, ψ, and α are pointwise feature transformations (e.g., an MLP) for generating the query, key, and value, respectively. Here, ρ(e.g., softmax) is the normalization function, and γ is a function that performs mapping (e.g., an MLP). In the Point Transformer framework [6], the relative position in each axis-direction is converted to a position encoding δ through an MLP. This position encoding δ is then added to both the attention score obtained through the subtraction relation and the value.

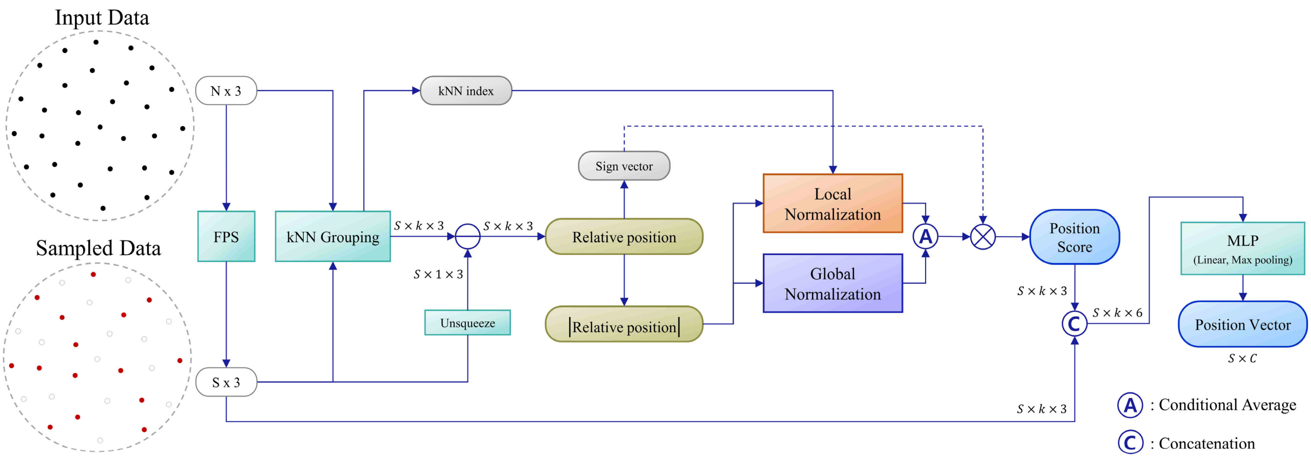

FIGURE 3. Generation of position vector through outlier-considered normalization in both local and global ranges, using the relative positions calculated within the kNN group.

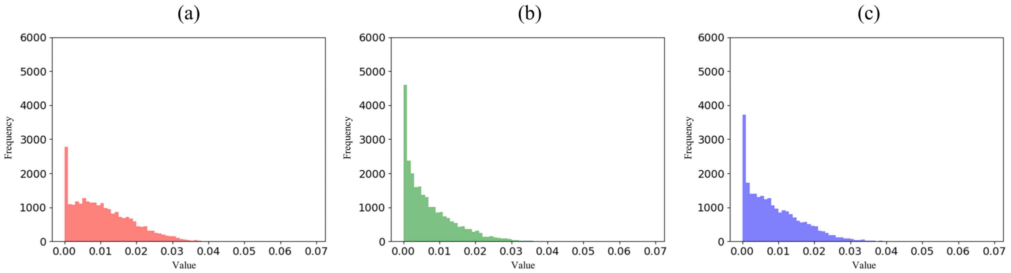

FIGURE 4. *x*-axis direction (a) , *y*-axis direction (b), and *z*-axis direction (c) values of the relative position within the kNN group. Compared with the values of the feature vector, which are typically between 0 and 1, the relative position values are extremely small.

4) POSITION VECTOR

This section describes the process of obtaining the position vector PV to be used in the PMHA (Fig. 3). The input data of the network are the absolute position information of the point cloud, denoted as S_N. The point cloud of S_N is halved through farthest point sampling to generate a sampled point cloud $S_S = \{p_{s1}, p_{s2}, \ldots, p_{sN}\}$. Next, based on point p_{si} in S_S, k-neighboring points from S_N are retrieved based on the 3D Euclidean distance to obtain a kNN group $S_K \in R^{N \times k \times 3}$. By subtracting p_{si} from S_K, the relative position ΔS_K is obtained, as follows:

$$\Delta S_K = S_K - p_{si}, \Delta S_K \in R^{N \times k \times 3} \tag{11}$$

The process of converting this relative position ΔS_K into a positional score suitable for point cloud data involves the following steps. Using the absolute value of the relative position ΔS_K, the local normalization $Norm_L$ and global normalization $Norm_G$ processes are implemented, as follows:

$$PS_L = Norm_L(|\Delta S_K|) \tag{12}$$

$$PS_G = Norm_G(|\Delta S_K|) \tag{13}$$

where PS_L is the result of $Norm_L$, and PS_G is the result of $Norm_G$. Section III-C provides details of the normalization used in $Norm_L$ and $Norm_G$.

Next, the conditional average of PS_L and PS_G is determined and multiplied by the original sign of the relative position to generate the position score PS. The PS is then concatenated with the absolute position S_N and input to the MLP to obtain the position vector PV. Section III-D describes the complete process from obtaining and merging PS_L and PS_G to generating the position vector PV.

C. NORMALIZATION OF RELATIVE POSITION

This section describes the min-max normalization of the relative position through interquartile range (IQR) outlier detection performed during the global and local normalizations (Fig. 3) to obtain the position vector. In processing point cloud data, the relative position of each point is pivotal in understanding the spatial relationships for deriving more accurate classification and segmentation results. However, several challenges arise when the relative position information is directly used.

First, the inter-point distance in each axis-direction, constituting the relative position, is significantly smaller than the

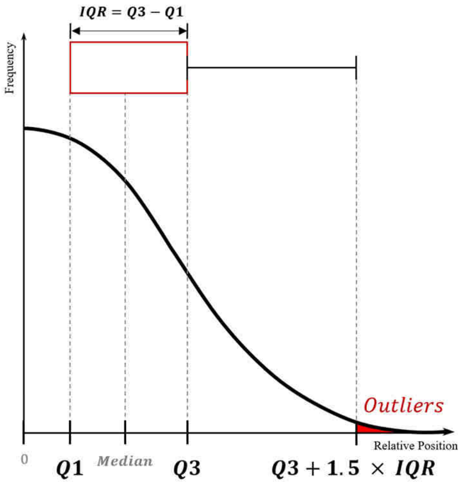

FIGURE 5. Stabilization of the value to be set as the maximum value before using min-max normalization, through IQR outlier detection.

feature vector values, which are generally between 0 and 1, as shown in Fig. 4. Consequently, the output becomes extremely small during the learning process, which involves multiplying by weights. Furthermore, the distance between even widely separated points rarely exceeds 0.03, which induces uncertainty in differentiating between nearby and distant points. Therefore, to meaningfully use the information of the nearby and distant points in the model, it is necessary to normalize the relative position, which is clustered around small values.

Second, points in close proximity tend to have similar meanings and strong correlations. Therefore, among the surrounding k points, those closer to the reference point are expected to exert a greater influence. However, the relative position approaches zero as the points become closer. In deep learning training, such values (approaching zero) mean that the model focuses less on the corresponding points.

Considering these aspects, we establish a positional encoding method that transforms relative positions into meaningful values reflecting the characteristics of point cloud data. The positional information generated through this method is normalized within the range of the object and kNN group based on the distance between points. The resulting position score has a total of three values in the x, y, and z-axis directions between two points.

1) SCALE OF RELATIVE POSITION

The positional information value must not be excessively large or small. If the value is too small, it may be difficult to extract meaningful information from the learning process. In contrast, if the value is too large, the influence of the feature information containing the meaning of the object diminishes, preventing proper learning. Therefore, normalization is performed to obtain a value between 0 and 1, inspired by the sine and cosine functions used in positional encoding in natural language processing.

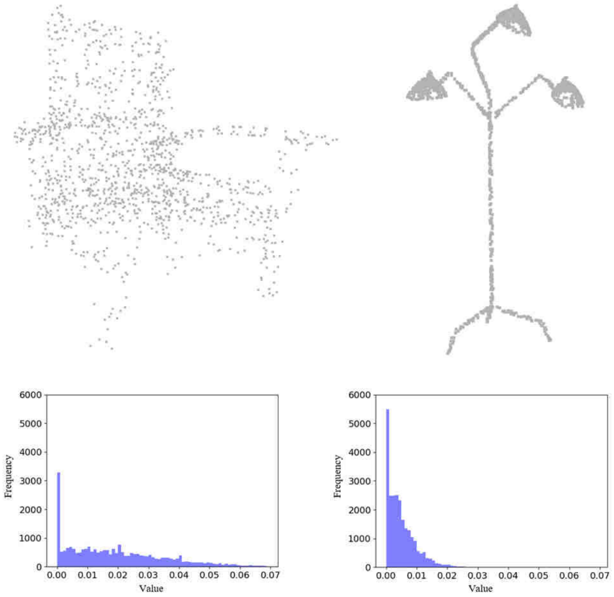

FIGURE 6. The point cloud of a chair is more distant between points compared to the point cloud of a lamp. The two objects exhibit differences in the degree of closeness and distance from each other.

2) CORRELATION OF POINTS

A smaller distance between points corresponds to a stronger correlation between them. Therefore, the position score is designed to approach 1 for nearby points and 0 for distant points. Because the score is generated at various point cloud resolutions throughout the network, it does not focus extensively on narrow areas.

The relative positions of the three axes are normalized considering the abovementioned aspects. During normalization, selection of the adjustment range of the data, handling of outliers, and variable-specific normalization must be carefully performed considering the data characteristics.

Using the min-max normalization strategy, normalization is independently performed for each of the three-dimensional axes. Because we consider the absolute value of the relative position, the minimum value is zero when min-max normalization is performed. This is because the reference point is included when calculating the relative position. A key step in this process is to determine the maximum value. In particular, when applying min-max normalization, it is important to prevent outliers that are much larger than the other data points from becoming the maximum value. Failure in removing the outliers may cause the original data pattern to be inaccurately reflected, with most of the scores being underestimated. To address this problem, we use the IQR outlier detection method shown in Fig. 5. This method identifies outliers based on the median of the data, which is more stable than the average as it is less susceptible to outliers.

The outlier threshold IQR_{th} is set using the IQR, representing the difference between the third quartile Q_3 and first quartile Q_1. Values exceeding this threshold are considered

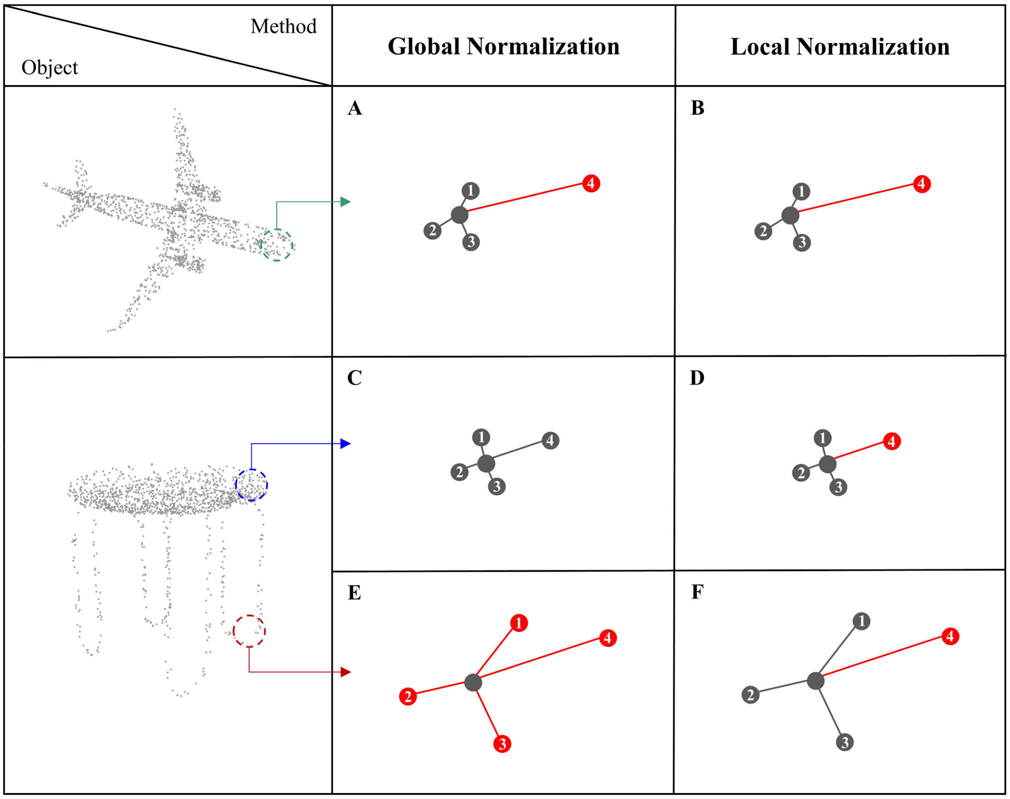

FIGURE 7. If normalization is only performed on the entire point cloud of each object (global Normalization), the points in the leg part of the chair would have low position scores.

outliers, as follows:

$$IQR = Q_3 - Q_1 \tag{14}$$

$$IQR_{th} = Q_3 + 1.5 \times IQR \tag{15}$$

By replacing all outliers with the threshold IQR_{th}, we set this threshold as the maximum value $\Delta S_{K,max}$ in the min-max normalization process. This allows us to perform normalization that is robust to outliers, as follows:

$$Outliers = IQR_{th} \tag{16}$$

$$\Delta S_{K,max} = IQR_{th} \tag{17}$$

Subsequently, min-max normalization is applied to the relative position, and the result is denoted as $\Delta S_{K,norm}$. Because the absolute value of the relative position is considered, the minimum value $\Delta S_{K,min}$ is 0, and the IQR_{th} becomes the $\Delta S_{K,max}$. Next, $\Delta S_{K,norm}$ is subtracted from 1 to ensure that small and large relative positions approach 1 and 0, respectively:

$$\Delta S_{K,norm} = 1 - \frac{\Delta S_K - \Delta S_{K,max}}{\Delta S_{K,max} - \Delta S_{K,min}} \tag{18}$$

Both global and local normalizations follow this normalization approach, with the only difference being the range over which normalization is performed, as discussed in the following text.

D. NORMALIZATION RANGE

1) GLOBAL NORMALIZATION

Our focus is on information regarding the distance, distribution, and orientation of other points relative to the reference point within a kNN group. Therefore, global normalization is first performed on all the relative positions obtained through grouping within a single object. Specifically, object-based normalization is performed because the density of the point cloud varies across objects given that objects of different sizes and surface areas are sampled with the same number of points. In Fig. 6, the relative positions of the points constituting the chair and lamp indicate that the criteria for being nearby and distant are different. Object-based normalization allows the network to reduce the bias stemming the density differences between objects and facilitates generalized learning. The three axis-direction values generated within the object range using Equations 14–18 are termed the global position score and denoted as PS_G.

2) LOCAL NORMALIZATION

As shown in Fig. 7, from the example of the chair, it can be observed that the density of point cloud components may vary even within a single object. In such cases, object-based global normalization may yield extremely low global position scores for low-density areas of the point cloud. Although the legs of the chair exhibit a low density when viewed in the context of the object, the density is consistent within the leg. Thus, these points should not be considered distant points. To address this issue, we perform local normalization within the kNN group to balance the position scores. The three axis-direction values generated through the same process as PS_G within the kNN range are termed the local position score and denoted as PS_L.

3) MERGE

This section describes the process for deriving the final score, i.e., the position score PS, from the global and local position scores obtained previously. Because the scale of positional information should not be excessively large or small, we compute the averages of the global and local position scores to obtain a score between 0 and 1. Nevertheless, it is necessary to address exceptions, given that objects may have varying point cloud densities across different parts.

Fig. 7 compares the results of outlier detection in both the global and local normalization ranges for an airplane point cloud with consistent density and a chair point cloud with different densities across parts. The points connected in red in the figure are outliers among the nearby k points. An examination of parts A and B in the figure shows that the airplane has a uniform density overall, and thus, the results of outlier detection are the same for both the global and local normalizations. Point 4 has a score of 0 because it is identified as an outlier in both normalization process, implying that this point is significantly unrelated to the reference point.

In contrast, for the chair point cloud, the densities of the seat and leg parts are considerably different, and thus, the outlier detection results vary with the normalization range.

Considering parts C and D in Fig. 7, point 4 appears as an outlier in local normalization but not in global normalization due to the influence of the sparse leg part. Consequently, according to Equation 20, point 4 achieves a score between 0 and 0.5 because the local position score is 0. In other words, although point 4 is not completely unrelated to the reference point within a dense local part owing to the presence of many nearby points, it is relatively distant compared with the remaining nearby points.

Observing parts E and F, point 4 is considered an outlier in both the global and local normalizations. Thus, its score is 0, implying that it has extremely low relevance with the reference point. In part E, global normalization is influenced by the dense seat part, and thus, points 1, 2, and 3 are considered outliers. However, in part F, points 1, 2, and 3 are not considered outliers in local normalization. In particular, given the point cloud of the chair, it is reasonable not to consider points 1, 2, and 3 as being distant from the reference point in F owing to the low density in the leg part. Therefore, according to Equation 19, if points considered outliers in global normalization are not identified as outliers in local normalization (e.g., points 1, 2, and 3 in F), only local normalization is applied to maintain the merged score between 0 and 1.

In summary, to achieve reasonable scoring for points such as 1, 2, and 3 in E and F, we design the system to perform normalization by merging the global position score PS_G and local position score PS_L as follows:

$$PS = PS_L(if\ PS_G = 0\ and\ PS_L \neq 0) \tag{19}$$

$$PS = \frac{PS_G + PS_L}{2}(otherwise) \tag{20}$$

where PS is the final score obtained by merging the position score in the global and local ranges.

Subsequently, to express directionality in accordance with the 3D characteristics of the point cloud, element-wise multiplication $\odot$ is performed with the position score PS, retaining the original sign of the relative position $sign$, as follows:

$$PS = PS \odot sign \tag{21}$$

In this case, points located at the same distance but in opposite directions exhibit the same absolute value of the score, with only the sign differing.

Lastly, to ensure that PV contains both absolute and relative positional information, we concatenate PS with $\mathrm{S_N}$. The MLPs ς, ϱ, and max pooling are applied to the position score to generate a position vector PV for use in multi-head attention, as follows:

$$PV = \varrho\,(maxpool\,(\varsigma\,(concat\,(PS; \mathrm{S_N})))) \tag{22}$$

E. UP-SAMPLING

The hierarchical network structure of PointNet++ [20], composed of down-sampling and up-sampling, has been used as the foundation for numerous point cloud segmentation networks. The up-sampling process identifies the closest three points in the next layer for each point in the previous layer and interpolates the feature vector based on the Euclidean distance. Despite ongoing research on down-sampling [6], [21], [22], [23], many recent studies have used the up-sampling process of PointNet++ as is [20].

To further research on up-sampling, we use the proposed position information. After interpolating the feature vector for the three neighboring points in the conventional method, we concatenate the position score and absolute position, and then pass them through the MLP to incorporate normalized positional information in the features to be delivered to the next layer. This addition of positional information results in a slight improvement in the segmentation performance, as observed in the ablation studies.

IV. EXPERIMENT

The effectiveness of the proposed network for part segmentation is evaluated using the public dataset, ShapeNet-Part [28], and its classification performance is evaluated using ScanObjectNN [29] and ModelNet40 [30]. Ablation studies are conducted to evaluate the part segmentation performance when specific components are removed. Furthermore, we aim to further the research on up-sampling in networks based on PointNet++ [20].

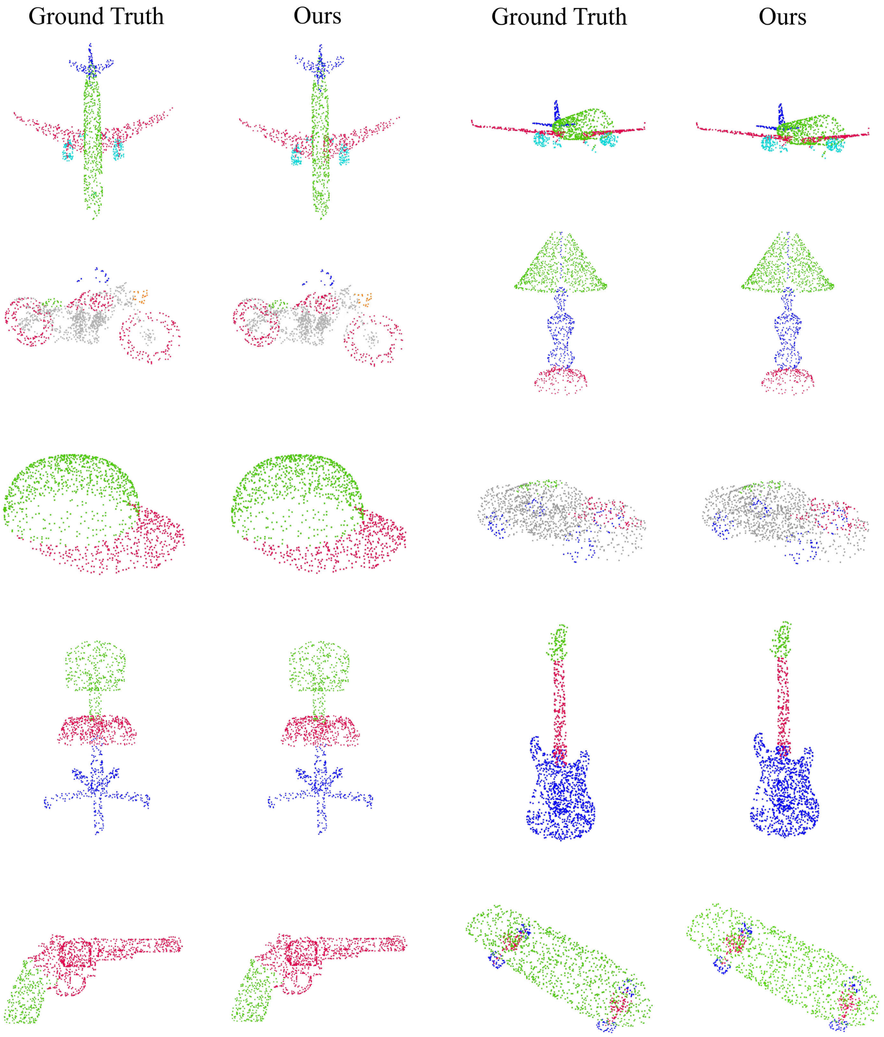

FIGURE 8. Visualization result of part segmentation in the ShapeNet-Part [28] dataset.

A. DATASET

Three datasets are used to evaluate the performance of the proposed approach in 3D shape classification and pointwise part segmentation tasks.

ShapeNet-Part [28] is a large-scale part segmentation dataset that includes 16,881 3D objects and over 50 part categories. This dataset provides pointwise 3D segmentation information for 16 shape classes, including objects such as cars, airplanes, and chairs, with each object divided into two or more parts. Consequently, this dataset is highly suitable for conducting part-wise segmentation and classification tasks on complex 3D shapes. The clear distinction of boundaries of each part [28] helps clarify the structural features of 3D objects.

The ScanObjectNN [29] classification dataset, based on real-world 3D scan data, categorizes 2,902 unique objects into 15 categories and includes 15,000 data entries. Each object includes attributes such as global and local coordinates, normals, colors, and semantic labels. Therefore, the ScanObjectNN [29] dataset is especially valuable for evaluating the model performance in real-world conditions and facilitates training in the presence of various problems that can occur in the real world, such as noise or occlusions. In this study, we select PB_T50_RS, considered the most challenging condition within the ScanObjectNN [29] dataset, as it involves location, rotation, and scale manipulations of objects. Such transformations enhance the understanding of the complex 3D shapes of objects and can help assess the robustness of a model against various transformation conditions encountered in the real world.

The ModelNet40 [30] classification dataset is based on computer-generated 3D CAD models. This dataset categorizes 12,311 data entries into 40 categories. The models are clean and accurately labeled, and each model corresponds to a standardized size and orientation. Thus, ModelNet40 [30] is primarily used to evaluate the performance of models focused on classification tasks.

TABLE 1. Part segmentation result on shapenet-part [28] dataset.

Methods	mIoU	air plane	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor bike	mug	pistol	rocket	skate board	table
PointNet [19]	83.7	83.4	78.7	82.5	74.9	89.6	73.0	91.5	85.9	80.8	95.3	65.2	93.0	81.2	57.9	72.8	80.6
PointNet++ [20]	85.1	82.4	79.0	87.7	77.3	90.8	71.8	91.0	85.9	83.7	85.3	71.6	94.1	81.3	58.7	76.4	82.6
Kd-Net [33]	82.3	80.1	74.6	74.3	70.3	88.6	73.5	90.2	81.2	71.0	94.9	57.4	86.7	78.1	51.8	69.9	80.3
SO-Net [34]	84.6	81.9	83.5	84.8	78.1	90.8	72.2	90.1	83.6	82.3	85.2	69.3	94.2	80.0	51.6	72.1	82.6
DGCNN [15]	85.2	84.0	83.4	86.7	77.8	90.6	74.7	91.2	87.5	82.8	95.7	66.3	94.9	81.1	63.5	74.5	82.6
SRN [35]	85.3	82.4	79.8	88.1	77.9	90.7	69.6	90.9	86.3	84.0	95.4	72.2	94.9	81.3	62.1	75.9	83.2
3D-GCN [36]	85.1	83.1	84.0	86.6	77.5	90.3	74.1	90.9	86.4	83.8	95.6	66.8	94.8	81.3	59.6	75.7	82.6
Point Trans [6]	86.6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Wang et al. [37]	85.3	82.2	82.0	84.4	78.8	90.5	77.9	91.2	86.7	83.4	95.8	74.8	94.1	81.8	63.1	75.6	82.0
DANet [38]	85.8	83.9	83.2	85.0	79.7	91.1	77.3	91.9	88.4	**84.8**	95.7	71.9	94.7	83.2	58.0	75.1	82.8
PointMLP [22]	86.1	83.5	83.4	87.5	80.5	90.3	**78.2**	92.2	88.1	82.6	96.2	**77.5**	**95.8**	85.4	64.6	**83.3**	**84.3**
PointStack [21]	87.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SPoTr [39]	87.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
LPGA [31]	85.7	82.3	81.0	**90.1**	79.2	**91.5**	71.3	90.8	**88.6**	84.4	96.0	70.3	94.8	81.9	64.2	76.1	82.5
Ours	**87.5**	**90.8**	**85.7**	68.7	**83.0**	90.1	70.0	**92.5**	88.4	79.6	**96.8**	71.2	93.5	**88.3**	**69.2**	69.8	82.9
Num of samples	-	2690	76	55	898	3746	69	787	392	1546	445	202	184	275	66	152	5263

TABLE 2. Classification result on scanobjectnn [29] and modelnet40 [30] dataset.

Methods	Year	ScanObjectNN [29]		ModelNet40 [30]	
		OA (%)	mAcc (%)	OA (%)	mAcc (%)
PointNet [19]	2017	68.2	63.4	89.2	86.0
PointNet++ [20]	2017	77.9	75.4	90.7	-
PointCNN [40]	2018	78.5	75.1	92.5	88.1
DGCNN [15]	2019	78.1	73.6	92.9	90.2
Point-PlaneNet [41]	2020	-	-	92.1	90.5
DRNet [42]	2021	80.3	78.0	93.1	-
GBNet [43]	2021	80.5	77.8	93.8	91.0
DynamicScale [44]	2021	82.0	-	92.1	-
CurveNet [45]	2021	-	-	93.8	-
Point-BERT [46]	2021	83.1	-	93.8	-
PRA-Net [47]	2021	82.1	79.1	93.7	91.2
LCPFormer [48]	2022	-	-	93.6	90.7
Point-TnT [49]	2022	83.5	81.0	92.6	-
PointMLP [22]	2022	85.7	84.4	**94.1**	**91.5**
PointStack [21]	2022	87.2	86.2	93.3	89.6
Point-PN [50]	2023	87.1	-	93.8	-
Ours		**87.6**	**86.6**	92.7	89.6

B. EVALUATION METRICS

Following previous studies [6], [31], [32], the performance of the model is evaluated using the following metrics:

For the ShapeNet-Part [28] dataset evaluations, we use the instance-wise mean intersection over union (ins. mIoU) metric, which measures the segmentation accuracy for each part. Ins. mIoU reflects the distribution of each instance within the dataset, treating each instance as an independent entity for evaluation. This helps minimize the impact of class imbalance on performance evaluation, counters classes with a

TABLE 3. **Ablation study on the components constituting pmha and the performance improvement from applying positional information in up-sampling.**

Normalization and Up-sampling	ScanObjectNN [29]		ShapeNet-Part [28]
	OA (%)	mAcc (%)	Inst. mIoU (%)
MHA	87.27	85.68	87.10
PMHA with V_0	87.23	85.84	87.08
PMHA with V_1	87.58	86.01	87.25
PMHA with V_2	**87.61**	**86.61**	87.38
PMHA with V_2 + Up-sampling	-	-	**87.50**

small number of instances from disproportionately affecting the overall model performance of the model, and prevents models from overly relying on classes with a large number of instances.

For the ScanObjectNN [29] and ModelNet40 [30] dataset evaluations, we use the mean accuracy *mAcc* and overall accuracy *OA* metrics. In particular, *mAcc* quantifies the average classification accuracy for each class, whereas *OA* indicates the ratio of accurate predictions among all predictions.

Four main components are considered for comparing predictions and labels: true positive *TP*, true negative *TN*, false positive *FP*, and false negative *FN*. The evaluation metrics are computed as follows:

$$OA = \frac{TP + TN}{TP + TN + FP + FN} \tag{23}$$

$$mAcc = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \tag{24}$$

$$mIoU = \frac{1}{C}\sum_{i=1}^{C}\frac{TP_i}{TP_i + FP_i + FN_i}, i \in C \tag{25}$$

where C is the number of classes in the dataset.

C. IMPLEMENTATION DETAILS

The network is implemented using Python 3.7 and PyTorch 1.9.1. Model training is performed using an NVIDIA RTX 3090ti (24 GB memory) system with CUDA 11.1.

The stochastic gradient descent optimizer is used for training, with a cosine annealing scheduler without warm restarts. The cross-entropy loss function with label smoothing is employed. The initial learning rate is set as 0.01. During training, if the learning rate drops below 0.0001, it is maintained at 0.0001.

For the ShapeNet-Part [28] dataset, training is performed for 400 epochs, using 2048 points per object, with a batch size of 8. For the ScanObjectNN [29] dataset, training is performed for 200 epochs, using 1024 points per object, with a batch size of 16. For the ModelNet40 [30] dataset, training is performed for 300 epochs, using 1024 points per object, with a batch size of 24.

D. EXPERIMENTAL RESULTS

1) PART SEGMENTATION

The proposed PMHA-Net is evaluated through a comparison with several other models over the ShapeNet-Part [28] dataset. As shown in Table 1, PMHA-Net achieves higher ins. mIoU scores than other state-of-the-art methods, demonstrating superior performance across various classes. The proposed approach, which effectively utilizes positional information, is particularly effective for classes with diverse and complex parts, such as airplanes and cars. Moreover, although our method yields lower scores for classes with less data, it generally outperforms other models in classes with sufficient data. This finding highlights the scope for improvement in classes with less data, either through the use of larger datasets or data augmentation techniques.

2) SHAPE CLASSIFICATION

The shape-classification performance of the proposed and existing approaches is evaluated over the ScanObjectNN [29] and ModelNet40 [30] datasets. As illustrated in Table 2, PMHA-Net outperforms the state-of-the-art methods on the ScanObjectNN [29] dataset. Although PMHA-Net achieves competitive performance on ModelNet40 [30], it does not outperform the existing models.

According to PointStack [21], this phenomenon may be attributable to the limited number of training samples available in ModelNet40 [30]. PointStack [21] constructed the ScanObjectNN [29] dataset to match the average number of training samples per class in ModelNet40 [30] and performed a validation exercise. In this experiment, while the model surpasses existing model with a sufficient dataset, it showed lower results compared to existing model when trained on an equally limited number of samples. In the case of ModelNet40 [30], 12,311 point clouds are provided as training data across 40 different classes. In contrast, the main-PB_T50_RS variant of ScanObjectNN [29] offers 15,000 point clouds for 15 classes, meaning it provides, on average, more training samples per class than ModelNet40 [30]. Given that the proposed model records lower scores for classes with fewer samples in the ShapeNet-Part [28] dataset (bottom rows of Table 2), it can be inferred that the limited number of training samples likely leads to the abovementioned observations.

E. ABLATION STUDIES

We have normalized the relative position in two ways and applied the position vector obtained through this to the multi-head attention. Additionally, we have designed a module that provides additional position information to the traditional up-sampling method. We performed an ablation study to verify the availability of each module and represented this in Table 3.

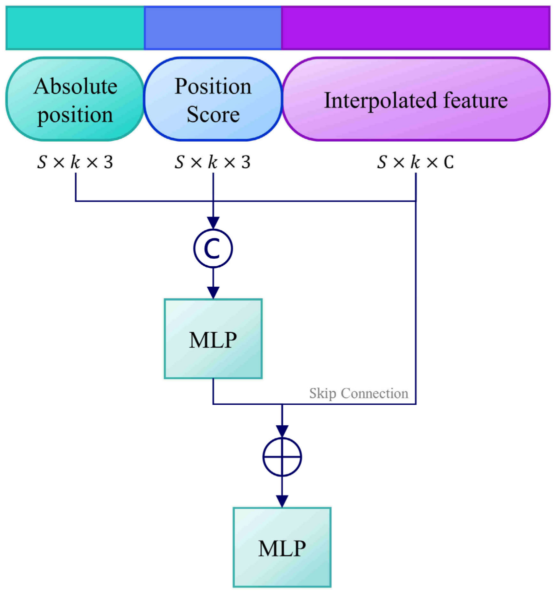

FIGURE 9. Positional up-sampling method using the result of the traditional interpolation method with three points (Interpolated feature).

1) NORMALIZATION

Through normalization of the relative position, we redesigned it as more suitable and meaningful information for the point cloud. In Table 3, MHA is an experimental result of multi-head attention that does not use positional information. PMHA with V_0 applies the position vector made with the relative position before normalization to the multi-head attention. Through comparison with MHA, it can be seen that the relative position, which is close to zero and gets smaller as the two points are closer, has little effect on performance and could even have a negative effect. PMHA with V_1 is the result of applying global normalization, while PMHA with V_2 is the result of adding local normalization to the global normalization. PMHA with V_2 increased the OA performance in classification by 0.34% and the mAcc performance by 0.93%, compared to MHA which does not utilize position information in multi-head attention, and the part segmentation performance increased by 0.32%.

2) UP-SAMPLING

Additional research is performed to further the exploration of the currently stagnant up-sampling method in the hierarchical structure based on PointNet++ [20]. Specifically, we obtain $F_{AL} \in \mathrm{R}^{\mathrm{N}} \times k \times C$ by concatenating the absolute position $S_{kNN} \in \mathrm{R}^{\mathrm{N}} \times k \times 3$ of each point within the kNN group to the interpolated feature $F_I \in \mathrm{R}^{\mathrm{N}} \times k \times C$ obtained by the existing up-sampling method and the position score $PS \in R^N \times k \times 3$ obtained from the relative position. The results are subjected to the MLP ϑ, as follows:

$$F_{AL} = \vartheta(concat\ (S_{kNN}; PS; F_I)) \tag{26}$$

In the up-sampling process, only global normalization is applied to generate the position score. This selection is made because only three points are grouped in up-sampling, and thus, it is not necessary to apply local normalization within the kNN group. Next, we obtain $F_U \in \mathrm{R}^{\mathrm{N}} \times C$ by skip connecting F_I to F_{AL} based on the dimension indicating the index of the surrounding k points and passing it through the MLP ζ, as follows:

$$F_U = \zeta(F_{AL} + F_I) \tag{27}$$

Through this process, the positional information in the up-sampling network can be reinforced. And we confirmed an improvement in the part segmentation performance on ShapeNet-Part [28], with the results at the PMHA with V_2 + Up-sampling in Table 3.

V. CONCLUSION AND FUTURE WORK

The proposed network includes PMHA, kNN grouping, and up-sampling. PMHA-Net is implemented at various point cloud resolutions. At each resolution, multiple heads are used to interpret feature and positional information from multiple perspectives. The kNN grouping method considers the 3D characteristics of the point cloud and distribution of each object to derive the position vector to be used in PMHA. In the up-sampling technique, positional encoding is realized by incorporating the position score within the interpolation process involving the surrounding three points. The proposed network exhibits outstanding performance on the ShapeNet-Part [28] dataset in the part segmentation task and competitive performance on the ScanObjectNN [29] and ModelNet40 [30] datasets in the classification task.

Future work can be aimed at leveraging the position vector in various multi-head attention methodologies. Additionally, we are attempting to develop a more robust position vector by efficiently using the grouping characteristics of the point cloud, along with absolute and relative positions. We are also exploring methodologies that utilize the relationships between previous and subsequent layers in the simple up-sampling process. The proposed network demonstrates potential for enhancing the performance in classes with a limited number of samples. To address dataset imbalances, additional training can be realized through data augmentation or the use of supplementary datasets.

REFERENCES

[1] J. H. Park, Y. E. Lim, J. H. Choi, and M. J. Hwang, "Trajectory-based 3D point cloud ROI determination methods for autonomous mobile robot," *IEEE Access*, vol. 11, pp. 8504–8522, 2023, doi: 10.1109/ACCESS.2023.3238824.

[2] G. Gao, H. Liu, and H. Yang, "Quality judgment of 3D face point cloud based on feature fusion," *IEEE Access*, vol. 10, pp. 106513–106519, 2022, doi: 10.1109/ACCESS.2022.3211082.

[3] S. Kim, J. Ha, and K. Jo, "Semantic point cloud-based adaptive multiple object detection and tracking for autonomous vehicles," *IEEE Access*, vol. 9, pp. 157550–157562, 2021, doi: 10.1109/ACCESS.2021.3130257.

[4] S. Ao, Q. Hu, B. Yang, A. Markham, and Y. Guo, "SpinNet: Learning a general surface descriptor for 3D point cloud registration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 11753–11762.

[5] H. Lee, J. Jeon, S. Hong, J. Kim, and J. Yoo, ''TransNet: Transformer-based point cloud sampling network,'' *Sensors*, vol. 23, no. 10, p. 4675, May 2023.

[6] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, ''Point transformer,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2021, pp. 16259–16268.

[7] D. Maturana and S. Scherer, ''VoxNet: A 3D convolutional neural network for real-time object recognition,'' in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2015, pp. 922–928.

[8] Y. Zhou and O. Tuzel, ''VoxelNet: End-to-end learning for point cloud based 3D object detection,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4490–4499.

[9] H.-Y. Meng, L. Gao, Y.-K. Lai, and D. Manocha, ''VV-Net: Voxel VAE net with group convolutions for point cloud segmentation,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 8500–8508.

[10] C. He, R. Li, S. Li, and L. Zhang, ''Voxel set transformer: A set-to-set approach to 3D object detection from point clouds,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8417–8427.

[11] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, ''Multi-view convolutional neural networks for 3D shape recognition,'' in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 945–953.

[12] H. Huang, E. Kalogerakis, S. Chaudhuri, D. Ceylan, V. G. Kim, and E. Yumer, ''Learning local shape descriptors from part correspondences with multiview convolutional networks,'' *ACM Trans. Graph.*, vol. 37, no. 1, p. 114, Jan. 2018.

[13] A. Dai and M. Nießner, ''3DMV: Joint 3D-multi-view prediction for 3D semantic scene segmentation,'' in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 452–468.

[14] X. He, S. Bai, J. Chu, and X. Bai, ''An improved multi-view convolutional neural network for 3D object retrieval,'' *IEEE Trans. Image Process.*, vol. 29, pp. 7917–7930, 2020.

[15] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, ''Dynamic graph CNN for learning on point clouds,'' *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, Oct. 2019.

[16] J. Liu, B. Ni, C. Li, J. Yang, and Q. Tian, ''Dynamic points agglomeration for hierarchical point sets learning,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 7546–7555.

[17] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, ''Graph neural networks for social recommendation,'' in *Proc. World Wide Web Conf.*, May 2019, pp. 417–426, doi: 10.1145/3308558.3313488.

[18] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, ''Graph neural networks in recommender systems: A survey,'' *ACM Comput. Surv.*, vol. 55, no. 5, pp. 1–37, May 2023, doi: 10.1145/3535101.

[19] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, ''PointNet: Deep learning on point sets for 3D classification and segmentation,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 77–85.

[20] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, ''PointNet++: Deep hierarchical feature learning on point sets in a metric space,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–14.

[21] K. T. Wijaya, D.-H. Paek, and S.-H. Kong, ''Advanced feature learning on point clouds using multi-resolution features and learnable pooling,'' 2022, *arXiv:2205.09962*.

[22] X. Ma, C. Qin, H. You, H. Ran, and Y. Fu, ''Rethinking network design and local geometry in point cloud: A simple residual MLP framework,'' 2022, *arXiv:2202.07123*.

[23] X. Lai, J. Liu, L. Jiang, L. Wang, H. Zhao, S. Liu, X. Qi, and J. Jia, ''Stratified transformer for 3D point cloud segmentation,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8500–8509.

[24] G. Qian, Y. Li, H. Peng, J. Mai, H. Hammoud, M. Elhoseiny, and B. Ghanem, ''PointNeXt: Revisiting PointNet++ with improved training and scaling strategies,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 23192–23204.

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, and I. Polosukhin, ''Attention is all you need,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–11.

[26] H. Zhao, J. Jia, and V. Koltun, ''Exploring self-attention for image recognition,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10076–10085.

[27] S. Ioffe and C. Szegedy, ''Batch normalization: Accelerating deep network training by reducing internal covariate shift,'' in *Proc. Int. Conf. Mach. Learn.*, Jun. 2015, pp. 448–456.

[28] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, ''ShapeNet: An information-rich 3D model repository,'' 2015, *arXiv:1512.03012*.

[29] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, ''Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1588–1597.

[30] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, ''3D ShapeNets: A deep representation for volumetric shapes,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1912–1920.

[31] A.-T. Tran, H.-S. Le, S.-H. Lee, and K.-R. Kwon, ''Local graph point attention network in point cloud segmentation,'' *IEEE Access*, vol. 11, pp. 33296–33312, 2023.

[32] Z. Tao, Y. Zhu, T. Wei, and S. Lin, ''Multi-head attentional point cloud classification and segmentation using strictly rotation-invariant representations,'' *IEEE Access*, vol. 9, pp. 71133–71144, 2021.

[33] R. Klokov and V. Lempitsky, ''Escape from cells: Deep kd-networks for the recognition of 3D point cloud models,'' in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 863–872.

[34] J. Li, B. M. Chen, and G. H. Lee, ''SO-Net: Self-organizing network for point cloud analysis,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9397–9406.

[35] Y. Duan, Y. Zheng, J. Lu, J. Zhou, and Q. Tian, ''Structural relational reasoning of point clouds,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 949–958.

[36] Z.-H. Lin, S.-Y. Huang, and Y.-C.-F. Wang, ''Convolution in the cloud: Learning deformable kernels in 3D graph convolution networks for point cloud analysis,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1800–1809.

[37] F. Wang, X. Wang, D. Lv, L. Zhou, and G. Shi, ''Separable self-attention mechanism for point cloud local and global feature modeling,'' *IEEE Access*, vol. 10, pp. 129823–129831, 2022.

[38] C. Zhou, Y. Xie, X. He, T. Yuan, and Q. Ling, ''Dual attention network for point cloud classification and segmentation,'' in *Proc. 41st Chin. Control Conf. (CCC)*, Jul. 2022, pp. 6482–6486.

[39] J. Park, S. Lee, S. Kim, Y. Xiong, and H. J. Kim, ''Self-positioning point-based transformer for point cloud understanding,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 21814–21823.

[40] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, ''PointCNN: Convolution on x-transformed points,'' in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 820–830.

[41] S. M. M. Peyghambarzadeh, F. Azizmalayeri, H. Khotanlou, and A. Salarpour, ''Point-PlaneNet: Plane kernel based convolutional neural network for point clouds analysis,'' *Digit. Signal Process.*, vol. 98, Mar. 2020, Art. no. 102633.

[42] S. Qiu, S. Anwar, and N. Barnes, ''Dense-resolution network for point cloud classification and segmentation,'' in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2021, pp. 3813–3822.

[43] S. Qiu, S. Anwar, and N. Barnes, ''Geometric back-projection network for point cloud classification,'' *IEEE Trans. Multimedia*, vol. 24, pp. 1943–1955, 2022.

[44] S. V. Sheshappanavar and C. Kambhamettu, ''Dynamic local geometry capture in 3D point cloud classification,'' in *Proc. IEEE 4th Int. Conf. Multimedia Inf. Process. Retr. (MIPR)*, Sep. 2021, pp. 158–164.

[45] T. Xiang, C. Zhang, Y. Song, J. Yu, and W. Cai, ''Walk in the cloud: Learning curves for point clouds shape analysis,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 915–924.

[46] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu, ''Point-BERT: Pre-training 3D point cloud transformers with masked point modeling,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 19313–19322.

[47] S. Cheng, X. Chen, X. He, Z. Liu, and X. Bai, ''PRA-Net: Point relation-aware network for 3D point cloud analysis,'' *IEEE Trans. Image Process.*, vol. 30, pp. 4436–4448, 2021.

[48] Z. Huang, Z. Zhao, B. Li, and J. Han, ''LCPFormer: Towards effective 3D point cloud analysis via local context propagation in transformers,'' *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 9, pp. 4985–4996, Sep. 2023.

[49] A. Berg, M. Oskarsson, and M. O'Connor, "Points to patches: Enabling the use of self-attention for 3D shape recognition," in *Proc. 26th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2022, pp. 528–534.

[50] R. Zhang, L. Wang, Z. Guo, Y. Wang, P. Gao, H. Li, and J. Shi, "Parameter is not all you need: Starting from non-parametric networks for 3D point cloud analysis," 2023, *arXiv:2303.08134*.

JAESEUNG JEON received the B.S. degree in automotive IT convergence from Kookmin University, Seoul, South Korea, in 2022. He is currently pursuing the M.S. degree with the Graduate School of Automotive Engineering and Researching, Intelligent Vehicle Signal Processing Laboratory. His research interests include deep learning, computer vision, and autonomous driving technologies.

SEOKJIN HONG received the B.S. degree in automotive IT convergence from Kookmin University, Seoul, South Korea, in 2022. He is currently pursuing the master's degree with the Graduate School of Automotive Engineering and Researching, Intelligent Vehicle Signal Processing Laboratory.

His research interests include deep learning, enhancement of GNN's performance, and point cloud and autonomous driving.

HOOKYUNG LEE received the M.S. degree from the Graduate School of Automotive Engineering, Kookmin University, Seoul, South Korea, in 2023. His research interests include point cloud and image deep learning, computer vision, and autonomous driving.

JEESU KIM (Member, IEEE) received the Ph.D. degree from the Department of Electrical Engineering, Pohang University of Science and Technology. He is currently an Assistant Professor with the Department of Cogno-Mechatronics Engineering, Pusan National University. His research interests include the development of 3D image processing and various types of signal processing techniques.

JINWOO YOO (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electrical engineering from the Pohang University of Science and Technology (POSTECH), in 2009, 2011, and 2015, respectively. He was a Senior Engineer with Samsung Electronics, from 2015 to 2019. He is currently an Associate Professor with the Department of Automobile and IT Convergence, College of Automotive Engineering, Kookmin University. His current research interests include autonomous driving technologies and signal/image processing techniques.