PointeNet

A lightweight framework for effective and efficient point cloud analysis

Gu, Lipeng; Yan, Xuefeng; Nan, Liangliang; Zhu, Dingkun; Chen, Honghua; Wang, Weiming; Wei, Mingqiang

DOI
10.1016/j.cagd.2024.102311

Publication date
2024

Document Version
Final published version

Published in
Computer Aided Geometric Design

Citation (APA)
Gu, L., Yan, X., Nan, L., Zhu, D., Chen, H., Wang, W., & Wei, M. (2024). PointeNet: A lightweight framework for effective and efficient point cloud analysis. *Computer Aided Geometric Design*, *110*, Article 102311. https://doi.org/10.1016/j.cagd.2024.102311

Comput. Aided Geom. Des. 110 (2024) 102311

Contents lists available at ScienceDirect

Computer Aided Geometric Design

journal homepage: www.elsevier.com/locate/cagd

PointeNet: A lightweight framework for effective and efficient point cloud analysis

Lipeng Gu [a,b], Xuefeng Yan [a,c,*], Liangliang Nan [d], Dingkun Zhu [e,**], Honghua Chen [a], Weiming Wang [f], Mingqiang Wei [a,b]

[a] School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, Jiangsu, China
[b] Shenzhen Research Institute, Nanjing University of Aeronautics and Astronautics, Shenzhen, 518038, Guangzhou, China
[c] Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, 210000, Jiangsu, China
[d] Faculty of Architecture and the Built Environment, Delft University of Technology, Delft, Netherlands
[e] School of Computer Science and Technology, Jiangsu University of Technology, Changzhou, 213001, Jiangsu, China
[f] Hong Kong Metropolitan University, Hong Kong, China

ARTICLE INFO

Keywords:
PointeNet
Point cloud analysis
Multivariate geometric encoding
Distance-aware semantic enhancement
3D object detection

ABSTRACT

The conventional wisdom in point cloud analysis predominantly explores 3D geometries. It is often achieved through the introduction of intricate learnable geometric extractors in the encoder or by deepening networks with repeated blocks. However, these methods contain a significant number of learnable parameters, resulting in substantial computational costs and imposing memory burdens on CPU/GPU. Moreover, they are primarily tailored for object-level point cloud classification and segmentation tasks, with limited extensions to crucial scene-level applications, such as autonomous driving. To this end, we introduce **PointeNet**, an **e**fficient **net**work designed specifically for **point** cloud analysis. PointeNet distinguishes itself with its lightweight architecture, low training cost, and plug-and-play capability, while also effectively capturing representative features. The network consists of a Multivariate Geometric Encoding (MGE) module and an *optional* Distance-aware Semantic Enhancement (DSE) module. MGE employs operations of sampling, grouping, pooling, and multivariate geometric aggregation to lightweightly capture and adaptively aggregate multivariate geometric features, providing a comprehensive depiction of 3D geometries. DSE, designed for real-world autonomous driving scenarios, enhances the semantic perception of point clouds, particularly for distant points. Our method demonstrates flexibility by seamlessly integrating with a classification/segmentation head or embedding into off-the-shelf 3D object detection networks, achieving notable performance improvements at a minimal cost. Extensive experiments on object-level datasets, including ModelNet40, ScanObjectNN, ShapeNetPart, and the scene-level dataset KITTI, demonstrate the superior performance of PointeNet over state-of-the-art methods in point cloud analysis. Notably, PointeNet outperforms PointMLP with significantly fewer parameters on ModelNet40, ScanObjectNN, and ShapeNetPart, and achieves a substantial improvement of over 2% in $3D\ AP_{R40}$ for PointRCNN on KITTI with a minimal parameter cost of 1.4 million. Code is publicly available at https://github.com/lipeng-gu/PointeNet.

* Corresponding author at: School of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, Jiangsu, China.
** Corresponding author.
E-mail addresses: gulp1224@nuaa.edu.cn (L. Gu), yxf@nuaa.edu.cn (X. Yan), Liangliang.Nan@tudelft.nl (L. Nan), zhudingkun@jsut.edu.cn (D. Zhu), chenhonghuacn@gmail.com (H. Chen), wmwang@hkmu.edu.hk (W. Wang), mingqiang.wei@gmail.com (M. Wei).

https://doi.org/10.1016/j.cagd.2024.102311

Available online 24 April 2024

1. Introduction

With the popularity of 3D sensors such as LiDARs and depth cameras Geiger et al. (2012), point cloud analysis has gained significant prominence in both academic research and industrial development Qi et al. (2017a,b); Li et al. (2018); Ma et al. (2022); Zhang et al. (2023c). Unlike grid-based RGB images, point clouds comprise unordered and irregular points that outline the surfaces of objects in 3D space. This characteristic poses significant challenges in designing effective point cloud analysis models.

PointNet and PointNet++ Qi et al. (2017a,b) are pioneering works capable of directly analyzing unordered point clouds without the need for preprocessing. Just as ResNet He et al. (2016) serves as a prominent backbone in image processing, PointNet/PointNet++ has emerged as the widely adopted backbone network in subsequent point cloud analysis models Qi et al. (2019); Yang et al. (2020); Shi et al. (2019); Ma et al. (2022). The iterative evolution in these methods adheres to the "*continual increment*" principle, primarily focusing on modifying the backbone network, PointNet++, to capture more representative local geometric features. Specific improvements fall mainly into two aspects: introducing well-designed local geometric extractors (such as Graph Convolution Wang et al. (2019); Lin et al. (2022), Adaptive Point Convolution Xu et al. (2021), and Residual Point Block Ma et al. (2022)) into the encoder or simply stacking repeated blocks to deepen the network Ma et al. (2022); Liu et al. (2019). Such strategies, while capable of boosting performance, inevitably lead to a notable surge in network parameters. Fortunately, Point-NN Zhang et al. (2023c) recognizes this longstanding challenge and for the first time attempts to employ a counterintuitive "*subtraction*" strategy by trimming PointNet++, retaining only its non-learnable components, i.e., sampling, grouping, and pooling. This allows tasks like point cloud classification, segmentation, and even 3D detection to be performed without the need for training. While this unique strategy significantly reduces training costs, it falls short of exploiting the local geometry properties of point clouds, relying solely on spatial neighboring information. Additionally, existing point cloud analysis methods, including Point-NN, primarily focus on object-level point clouds, with limited extensions to more valuable scene point clouds (such as those in autonomous driving scenarios). These problems hinder Point-NN from fully unleashing its potential.

In this paper, we propose an **e**fficient **net**work for ***point*** cloud analysis, dubbed ***PointeNet***. Our model is highly streamlined, comprising only non-parametric components such as sampling, grouping, pooling, and a minimal number of learnable parameters, and can be flexibly combined with a point cloud classification/segmentation head, or be embedded into cutting-edge 3D detection networks tailored for real-world autonomous driving scenarios to enhance performance. Specifically, PointeNet comprises two crucial modules: the Multivariate Geometric Encoding (MGE) module, and the Distance-aware Semantic Enhancement (DSE) module. MGE comprises non-parametric farthest point sampling (FPS), k-nearest neighbor (k-NN), and pooling components, along with our proposed Multivariate Local Geometric Aggregation (MLGA) module incorporating a minimal number of Multi-Layer Perceptron (MLP) layers. This module captures multivariate 3D geometric features within local regions of point clouds, encompassing curvature, normal, and spatial neighboring information. DSE is a lightweight module specifically tailored for autonomous driving scenarios, incorporating only two MLPs and four Fully Connected (FC) layers. It focuses on capturing the rich semantic features of the point cloud through 3D semantic segmentation tasks, decorating the point cloud to significantly enhance the performance of arbitrary 3D detection networks. Notably, these semantic features are distance-aware, allowing for dynamic adjustment of the segmentation difficulty for each point, particularly focusing more "attention" on challenging distant points. We evaluate the performance of PointeNet through a comparative analysis with thirty-three competitors across the ScanObjectNN Uy et al. (2019), ModelNet40 Wu et al. (2015), ShapeNetPart Yi et al. (2016), and KITTI Geiger et al. (2012) datasets. Remarkably, PointeNet surpasses all competitors, showcasing its superior performance. The contributions of this work are threefold:

- We propose an efficient network, dubbed PointeNet. It excels in lightweight capture and adaptive aggregation of multivariate geometric and semantic features, making it suitable for point cloud segmentation/classification and enhancing cutting-edge 3D object detection networks tailored for autonomous driving scenarios.
- We propose a lightweight Multivariate Geometric Encoding (MGE) module, featuring non-learnable FPS, k-NN, and pooling components, supplemented by a Multivariate Local Geometric Aggregation (MLGA) module with a minimal number of learnable parameters, to capture diverse 3D geometric features.
- We propose an optional lightweight Distance-aware Semantic Enhancement (DSE) module tailored for autonomous driving scenarios. It dynamically adjusts the segmentation difficulty for each point, giving more "attention" to challenging distant points to capture distance-aware semantic features.

2. Related work

2.1. Point cloud analysis

There are two main paradigms for processing irregular and unordered point clouds: grid-based and point-based methods. Grid-based methods Liang et al. (2022); Qi et al. (2019); Shi et al. (2020); Lang et al. (2019) first project irregular point clouds onto regular grids, such as pillars or voxels, and then process them using 2D/3D convolutional neural networks (CNNs). This paradigm significantly enhances the processing speed by transforming the point cloud into structured grids. However, projecting onto grids may lead to information loss, degrading the quality of the point cloud representation Yang et al. (2019).

In contrast, point-based methods have emerged to directly process irregular point clouds without additional regularization, preserving the point cloud details more accurately. PointNet Qi et al. (2017a) is a pioneering method that utilizes shared MLPs to handle unordered point clouds as input directly. PointNet++ Qi et al. (2017b) builds upon PointNet by introducing a hierarchical feature

learning paradigm, allowing for the recursive capture of local geometric structures. Due to the promising performance exhibited by PointNet++, particularly in leveraging local geometric representations, including multi-scale geometric information, it has become a foundational element in modern point cloud analysis methods. Subsequent works, such as KPConv Thomas et al. (2019), RSCNN Liu et al. (2019), 3D-GCN Lin et al. (2022), PAConv Xu et al. (2021), PointConv Wu et al. (2019) and PointMLP Ma et al. (2022), either introduce intricate local geometric extractors or simply stack repeated network blocks to achieve further performance improvements. However, these strategies make the networks more cumbersome and less efficient. Subsequently, Point-NN adopts a contrary approach by abandoning intricate geometric extractors and relying solely on FPS, k-NN, and pooling to extract neighborhood information, achieving commendable performance. However, due to its capture of geometric features being singular and simplistic, it still falls short of achieving state-of-the-art (SOTA) performance. *Drawing on the successful experience of Point-NN, we propose PointeNet, an efficient network primarily based on these non-learnable components, aiming to capture representative multivariate geometric and semantic features at a minimal parameter cost.*

2.2. Local geometry exploration

Since the powerful ability to capture local geometric features of point clouds demonstrated by PointNet++ Qi et al. (2017b), subsequent research has mainly focused on exploring local geometric representations, which can be divided into three categories: convolution-based, graph-based, and attention-based methods. Classic convolution-based methods, with PointConv Wu et al. (2019) as a representative example, employ MLPs to approximate continuous weights and density functions within convolution filters. It extends dynamic filtering to a novel convolution operation. Unlike convolution-based methods, graph-based methods explore relationships between points. For instance, DGCNN Wang et al. (2019) introduces EdgeConv, a novel method that generates edge features describing the relationships between points and their neighbors, capturing local features of the point cloud. 3D-GCN Lin et al. (2022) employs a 3D graph convolutional network to form deformable 3D kernels, directly performing convolutional computations on point clouds. Related to graph-based methods, attention-based methods similarly focus on exploring direct relationships between points, as seen in works like PCT Guo et al. (2021) and Point Transformer Zhao et al. (2021). Although these methods have achieved success in point cloud analysis, they have consistently employed an "*increment*" principle, undoubtedly introducing more learnable parameters and thereby reducing computational efficiency and exacerbating the burden on CPU/GPU. Compared with these methods, our PointeNet achieves better performance with fewer parameters.

2.3. Deep network architecture for point clouds

The advancement of point cloud analysis methods is often inspired by breakthroughs in image processing networks. For instance, following the success of ResNet He et al. (2016) based on simple convolutions in the field of image processing, subsequent methods often use it as a backbone and continuously improve upon it. Similarly, after the breakthrough achieved by PointNet/PointNet++ based on simple MLPs in point cloud processing, they have become the mainstream backbones for point cloud analysis. After the success of graph- Felzenszwalb and Huttenlocher (2004), attention- Wang et al. (2018), and transformer-based Dosovitskiy et al. (2021) methods in the field of image processing, they have inspired a series of point cloud analysis works Wu et al. (2019); Wang et al. (2019); Ran et al. (2021); Guo et al. (2021); Zhao et al. (2021). For example, both Stratified-Transformer Lai et al. (2022) and 3DCTN Lu et al. (2022) are based on Transformers. The former significantly expands the receptive field through a stratification strategy, thereby efficiently capturing long-range context, while the latter combines the powerful local feature learning ability of convolutions with the remarkable global context modeling capability of Transformers. Although effective, transformers lead to an increase in the number of parameters. Moreover, PointNeXt Qian et al. (2022) explores the potential of PointNet++ via training strategy and model scaling. However, experimental evidence suggests that its significant performance improvements in some scenarios largely depend on multiple GPUs to increase the batch size (e.g., using four GPUs, each with a batch size of 8, on the ShapeNetPart dataset) during training, thus necessitating substantial hardware resources. Differently, our PointeNet has fewer parameters and does not require a large batch size during training, yet achieves state-of-the-art performance.

3. Revisiting point-based methods

Unlike grid-based methods, point-based methods have gained popularity due to their ability to directly learn the underlying point cloud representation without preprocessing. PointNet/PointNet++ Qi et al. (2017a,b) are pioneers in this paradigm, employing a hierarchical feature learning approach through the stacking of multiple learning stages. We first delve into the core idea of PointNet++ and then explore the strengths and weaknesses of subsequent methods.

Given a set of points $\mathcal{P} = \left\{ p_i \mid i = 1, \ldots, \mathcal{N} \right\} \in \mathbb{R}^{\mathcal{N} \times 3}$, where $\mathcal{N}$ is the number of points. The core idea of PointNet++ is to utilize FPS for down-sampling point clouds, use k-NN for grouping points, and then employ local geometric extractors to extract features, followed by max pooling to aggregate these features. The key steps for geometric feature extraction and aggregation can be formulated as:

$$f_c = \mathcal{A}\left(\Phi\left(f_{c,j} \right) \mid j \in \mathcal{N}_c \right) \tag{1}$$

where $\mathcal{A}(\cdot)$ means aggregation function (i.e., max-pooling in PointNet++), $\Phi(\cdot)$ denotes the local feature extractor (i.e., MLPs in PointNet++), and $f_{c,j}$ is the j-th neighbor point feature of center point p_c within the local region.

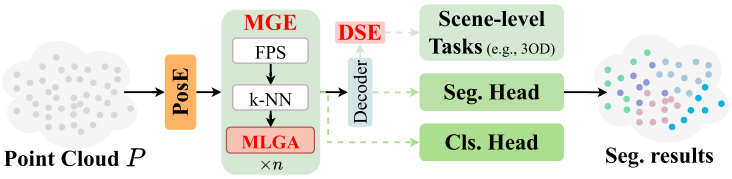

Fig. 1. Overview of PointeNet. It is an efficient network comprising two key modules: the Multivariate Geometric Encoding (MGE) module and the Distance-aware Semantic Enhancement (DSE) module. MGE captures multivariate geometric features, while DSE captures distance-aware semantic features. It can be flexibly combined with segmentation/classification heads to achieve excellent point cloud segmentation/classification tasks. Furthermore, it can be embedded in arbitrary scene-level tasks, such as 3D object detection (3OD), thereby further enhancing their performance.

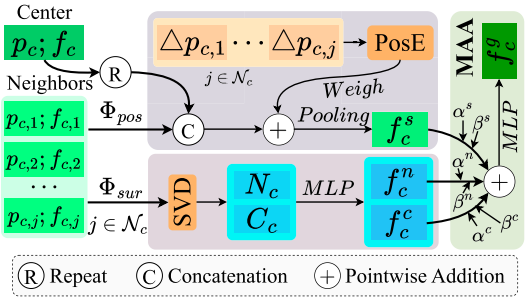

Fig. 2. Overview of MLGA. It is capable of lightweight capture of 3D geometries, including the spatial neighboring f_c^s, curvature f_c^c, and normal f_c^n features within a local region, and then undergoes a multivariate adaptive aggregation (MAA) module to adaptively aggregate various features, outputting multivariate local geometric features f_c^g.

Regarding network architecture design, PointNet++ establishes a versatile baseline for point cloud analysis. Later, based on this baseline, several methods are introduced, either emphasizing local geometric extractors Wang et al. (2019); Lin et al. (2022); Thomas et al. (2019); Xu et al. (2021) or stacking repeated network blocks to deepen the network Ma et al. (2022); Liu et al. (2019). While these methods can capture detailed local geometric information and often yield promising performance, their development encounters three challenges.

Firstly, the incorporation of complex geometric extractors or the expansion of the network to improve the representation of local geometric features invariably introduces a significant number of learnable parameters. This increase leads to higher computational complexity and places a greater strain on CPU/GPU resources. For example, PointMLP Ma et al. (2022) stacks residual network blocks to reach 16.7 million parameters, and training for segmentation tasks on ShapeNetPart can take up to 23 hours. This demonstrates a substantial computational burden. *Secondly*, the development of intricate local geometric extractors is approaching maturity and, in some cases, saturation. This is evidenced by the growing difficulty in securing performance improvements on well-known benchmarks like ModelNet40, ScanObjectNN, and ShapeNetPart. *Thirdly*, current methods are predominantly tailored for object-level point cloud analysis scenarios and rarely consider more realistic and valuable scene-level applications, such as autonomous driving. These limitations prompt us to explore a new paradigm that "lightens the load" on deep learnable networks. This is, *is it feasible to devise an efficient, lightweight framework for point cloud analysis that not only requires fewer parameters but also delivers satisfactory performance at both the object and scene levels?*

Excitingly, Point-NN has made the first successful attempt to perform point cloud analysis by constructing a PointNet++-style hierarchical framework that relies solely on non-learnable components such as FPS, k-NN, and pooling. However, it depends only on single spatial neighborhood information, overlooking other potential point cloud features, such as normals and curvatures of local surfaces, as well as semantic features. Moreover, it lacks customized optimization for real-world, scene-level applications.

4. Methodology

4.1. Overview

To overcome the aforementioned limitations, we introduce PointeNet (see Fig. 1), an efficient network designed for point cloud analysis. This network is adept at lightweightly capturing and adaptively aggregating diverse geometric and semantic features of point clouds with a minimal number of learnable parameters, ensuring a lightweight yet potent processing method. Additionally, it is specifically optimized for real-world autonomous driving scenarios.

The process of encoding multivariate geometric features (see Fig. 2) and distance-aware semantic features (see Fig. 3) can be formulated as:

$$f_c^g = \mathcal{A}\left(\Phi_{\text{pos}}\left(\left\{\left(p_{c,j}, f_{c,j}\right)\right\}_{j\in\mathcal{N}_c}, \left(p_c, f_c\right)\right), \Phi_{\text{sur}}\left(\left\{p_{c,j}\right\}_{j\in\mathcal{N}_c}\right)\right) \tag{2}$$

$$f_c^s = \Phi_{sem}\left(p_c, f_c^g\right) \tag{3}$$

where $\left(p_{c,j}, f_{c,j}\right)$ represent the coordinates and features of the j-th neighbor point of the center point $\left(p_c, f_c\right)$ within the local region, respectively. $\Phi_{pos}(\cdot)$ and $\Phi_{sur}(\cdot)$ are functions responsible for capturing spatial neighboring features and curvature-normal features, respectively. And, $\Phi_{sem}(\cdot)$ is a function for extracting distance-aware semantic features tailored for real-world autonomous driving scenarios. $\mathcal{A}$ is an aggregation function, and in PointeNet, it is our proposed multivariate adaptive aggregation (MAA) module.

Our efficient PointeNet exhibits some prominent advantages: i) PointeNet inherently exhibits permutation invariance, which aligns seamlessly with the properties of point clouds. ii) PointeNet bypasses the necessity for meticulously crafted feature extractors, effectively accomplishing position encoding, multivariate feature extraction, and feature aggregation with a minimal set of learnable parameters. iii) Due to its streamlined network architecture, PointeNet can be trained without the need for substantial hardware resources, such as multiple GPUs.

4.2. Multivariate geometric encoding

To accurately represent a point cloud, it is intuitive to aim for a comprehensive understanding of its spatial relationships, surface characteristics, and semantic attributes. Nevertheless, current methods (e.g., PointMLP Ma et al. (2022) and Point-NN Zhang et al. (2023c)) largely neglect surface intricacies or rely on intricate fusion methods to blend diverse features. Addressing this gap, we propose the Multivariate Geometric Encoding (MGE) module, aimed at efficiently capturing and adaptively aggregating multivariate geometric features.

MGE includes non-parametric FPS, k-NN, pooling components, and the Multivariate Local Geometric Aggregation (MLGA) module, with MLGA serving as the crucial module. As shown in Fig. 2, MLGA is capable of capturing spatial neighboring information within local regions, as well as comprehensively detailing local surface features, including curvature and normal information. Additionally, it facilitates the adaptive aggregation of these multivariate geometric features. Note that MLGA incorporates only a minimal number of MLPs for the transformation of feature channels. For example, PointeNet involves significantly fewer parameters compared to PointMLP: only 8.73% (1.1 million vs. 12.6 million) on ModelNet40, 7.14% (0.9 million vs. 12.6 million) on ScanObjectNN, and 51.88% (8.3 million vs. 16.0 million) on ShapeNetPart. In the following, we detail the process of capturing and aggregating multivariate geometric features.

The encoding of spatial neighboring information, similar to Point-NN, is represented by the following formula:

$$f_c^s = \left(C\left(f_c, f_{c,j}\right) + PosE\left(\triangle p_{c,j}\right)\right) \odot PosE\left(\triangle p_{c,j}\right), j \in \mathcal{N}_c \tag{4}$$

where f_c and f_j represent the center and neighbor features within the local region, respectively. $\triangle p_{c,j}$ represents the normalized coordinates of neighboring points by the mean and standard deviation. $PosE(\cdot)$ refers to the position encoding, and here, it is implemented using trigonometric functions.

Surface feature information in point clouds refers to a set of features that represent the shape of surfaces in 3D space. These features are crucial for point cloud analysis, with common surface feature information including normals, curvature, smoothness, boundaries, and even textures. Among these, normals and curvature are the most representative surface features. GeoMAE Tian et al. (2023) has successfully demonstrated this by using normals and curvature of local surfaces in point clouds as self-supervised signals for pre-training, thereby enhancing the performance of downstream tasks such as 3D object detection. Therefore, to improve the representation of single spatial neighboring features in the encoder, we incorporate surface curvature and normal as two additional geometrical features. For a set of neighboring points p_j, we initiate the process by calculating the matrix:

$$M_c = \frac{1}{\mathcal{N}_c} \sum_{j=1}^{\mathcal{N}_c} p_{c,j} p_{c,j}^T - \bar{p}_c \bar{p}_c^T \tag{5}$$

where $\bar{p}_c = \frac{1}{\mathcal{N}_c} \sum_{j=1}^{\mathcal{N}_c} p_{c,j}$ is the centroid of the local region. Next, we perform Singular Value Decomposition (SVD) on the matrix M_c to obtain the eigenvalues c_1, c_2, c_3 and corresponding eigenvectors n_1, n_2, n_3. Here, we use the eigenvector n_3 corresponding to the smallest eigenvalue λ_3 as the pseudo-normal vector $N_c = n_3$. Following that, we normalize the three eigenvalues to obtain the pseudo-curvature vector $C_c = \{c_0, c_1, c_2\}$:

$$c_m = \frac{c_m}{\sum_{i=1}^{3} c_i}, m \in \{1, 2, 3\} \tag{6}$$

As both pseudo-curvature and pseudo-normal vectors are low-dimensional, it is necessary to embed them into high-dimensional geometric features using an MLP layer, i.e., $f_c^n = MLP\left(N_c\right)$ and $f_c^c = MLP\left(C_c\right)$.

Unlike previous methods Zhang et al. (2019) that aggregate multiple features using several MLPs, we ingeniously introduce learnable parameters α, β to efficiently learn channel-wise weights and biases for each feature with fewer parameters, i.e., f_c^s, f_c^n and f_c^c:

$$f_c^g = \sum_{i=\{s,c,n\}} \left(\alpha^i \odot f_c^i + \beta^i\right) \tag{7}$$

where $\odot$ denotes dot product and f_c^g is the output of aggregated multivariate geometric features. These multivariate geometric features, after being processed through MLP layers to expand feature channels, achieve higher robustness by adaptively aggregating

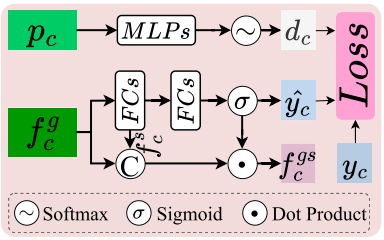

Fig. 3. Overview of DSE. It introduces the distance factor d into the semantic segmentation loss function, adaptively differentiating segmentation difficulty by region to derive distance-aware semantic features f_c^s. These features are combined with the geometric features f_c^g to output the final features f_c^{gs}.

using learnable parameters and incorporating data augmentation strategies (e.g., random translations, scaling, and rotations) during the training process.

4.3. Distance-aware semantic enhancement

Prior point cloud analysis methods, such as PointMLP Ma et al. (2022) and Point-NN Zhang et al. (2023c), are traditionally limited to object-level classification/segmentation tasks and lack custom optimizations for scene-level tasks, such as applications in autonomous driving scenarios. In real-world autonomous driving applications, the field of view is significantly broader, and point clouds tend to become sparser with increasing distance. This sparsity poses a challenge for effective perception, especially for distant points in scene-level tasks. Differently, object-level tasks involve dealing with a single object, which is limited in size and features a relatively uniform point cloud distribution, thus avoiding problems of varying densities between distant and nearby point clouds.

The intuitive solution is to allocate more "attention" to the sparser point clouds at a distance, while the relatively denser point clouds nearby require only a small amount of "attention" for effective perception. PV-RCNN Shi et al. (2020) aims to highlight foreground points through semantic segmentation, yet it overlooks the varying segmentation difficulties present in different areas of the point cloud scene, especially those differences related to distance. Consequently, the essential strategy is to reallocate attention from the denser, nearby point clouds to the sparser, distant regions, facilitating a more balanced perception network.

To address it, we introduce the Distance-aware Semantic Enhancement (DSE) module (see Fig. 3), which incorporates the distance factor into the semantic segmentation task (using Focal Loss Lin et al. (2020) for segmentation loss). It encodes categorical semantic information for each point as semantic features to enhance the semantic perception of point clouds, particularly for distant points. Specifically, DSE employs just four fully connected (FC) layers and the *Sigmoid* function, based on the multivariate geometric features f^g from MGE and corresponding position p_c, to perform binary classification for each point (foreground or background) and outputs the segmentation features from the intermediate segmentation process as semantic features f^s. These features contain rich semantic details about whether each point is foreground or background, and can serve as additional features of the point cloud to facilitate the network to better perceive foreground objects. Additionally, it utilizes two MLPs and the *Softmax* function to extract the distance factor $d \in [0, 1]$ based on the absolute values of the x, y coordinates for each point p. Owing to its architecture, comprising just four FCs and two MLPs, DSE stands out for its exceptional lightness, with a minimal parameter cost of only 0.017 million.

Subsequently, we incorporate d_c of each point into Focal Loss Lin et al. (2020), replacing the original adjustment factors α and γ to adaptively adjust the segmentation penalty based on the distance, as expressed by the formula:

$$L_{seg} = -d_c \left(1 - p_t\right)^{\frac{1}{d_c}} \log p_t \tag{8}$$

where $p_t = \left(1 - y_c\right) * \left(1 - \hat{y}_c\right) + y_c * \hat{y}_c$ signifies the probability that a predicted sample is classified as the positive class. The term $-d_c \left(1 - p_t\right)^{\frac{1}{d_c}}$ functions as the weight for the $\log p_t$ term, with d_c serving as an adaptive factor for each point that adjusts this weight. For distant points, their x, y coordinates are relatively large, yielding higher distance factors d_c. Conversely, for closer points, their distance factors d_c are smaller. As a result, $\frac{1}{d_c}$ for distant points is smaller than for nearby points, resulting in a greater penalty for distant points represented by $\left(1 - p_t\right)^{\frac{1}{d_c}} \log p_t$ compared to nearby points. Finally, the overall loss is scaled by the distance factor d_c.

To enrich point clouds with pointwise features f^{gs} that capture more information, we concatenate multivariate geometric features with distance-aware semantic features and then multiply this combination by the predicted probability of each point, denoted as $f^{gs} = C\left(f_c^g, f_c^s\right) * \hat{y}_c$. Concatenating these two features directly can better preserve both geometric and semantic information, and multiplying them by the predicted probabilities can more effectively highlight foreground points. By decorating the point cloud with additional representative features, necessitating only minor code adjustments and a low computational expense (1.4 million parameters), the efficiency of any 3D object detection network can be significantly boosted cost-effectively and efficiently.

5. Experiments

We conduct a comprehensive evaluation of PointeNet not only on object-level datasets such as ModelNet40 Wu et al. (2015), ScanObjectNN Uy et al. (2019), and ShapeNetPart Yi et al. (2016), but also on the scene-level dataset KITTI Geiger et al. (2012).

Table 1
Shape classification on synthetic ModelNet40. We report the class-average accuracy (mAcc) and overall accuracy (OA) on the testing set. The best performance value is in **bold**, and the second-best is underlined. Note that C denotes the embedding dimension of PointNet and $*$ means the reproduced results.

Method	mAcc (%)	OA (%)	Param.	FLOPs	Train Time
PointNet	86.0	89.2	-	-	-
PointNet++	-	91.9	1.4 M	-	-
PointCNN	88.1	92.5	-	-	-
PointConv	-	92.5	18.6 M	-	-
KPConv	-	92.9	15.2 M	-	-
Point Transformer	90.6	93.7	-	-	-
EQ-PointNet++	-	93.2	-	-	-
3DCTN (Multi-scale)	**91.6**	93.2	4.2 M	3.7 G	-
PointNeXt-S	90.8	93.2	1.4 M	1.6 G	-
APES	-	93.8	-	-	-
FlatNet-P	-	92.6	-	-	-
PointMLP*	90.8	93.3	12.6 M	14.6 G	6.2 h
Point-NN*	-	81.3	0 M	0 G	0 h
PointNet (C = 36) (Ours)	91.5	**93.9**	1.1 M	2.0 G	2.4 h
PointNet (C = 66) (Ours)	91.4	**93.9**	3.2 M	6.7 G	4.4 h

Detailed ablation studies demonstrate the effectiveness and efficiency of PointNet with both quantitative and qualitative analysis. Note that our experiments are executed on a single RTX 4090 GPU.

5.1. Datasets

For *Shape Classification*, we evaluate the performance on the synthetic ModelNet40 Wu et al. (2015) and the real-world ScanObjectNN Uy et al. (2019) datasets. ModelNet40 comprises 9843 training and 2468 testing meshed CAD models distributed across 40 categories. ScanObjectNN includes 15000 objects categorized into 15 classes, featuring 2902 unique object instances in real-world scenarios. Due to the presence of background, noise, and occlusions, ScanObjectNN poses significant challenges for existing methods. In our experiments, we specifically consider the most challenging perturbed variant (PB_T50_RS).

For *Part Segmentation*, we employ the ShapeNetPart Yi et al. (2016) dataset, which consists of 16881 shapes with 16 classes, totaling 50 part labels. In each class, the number of parts varies from 2 to 6.

For *3D Object Detection*, our experiments are conducted on the KITTI dataset Geiger et al. (2012), which is a popular benchmark for 3OD in autonomous driving scenarios. It contains 7481 samples for training and 7518 samples for testing. Each sample consists of a point cloud and an RGB image with nine categories.

5.2. Shape classification on ModelNet40

We also evaluate PointNet and compare our method with several recent works, including PointNet Qi et al. (2017a), PointNet++ Qi et al. (2017b), PointCNN Li et al. (2018), PointConv Wu et al. (2019), KPConv Thomas et al. (2019), Point Transformer Zhao et al. (2021), EQ-PointNet++ Yang et al. (2022), 3DCTN Lu et al. (2022), PointNeXt Qian et al. (2022), APES Wu et al. (2023), FlatNet Zhang et al. (2023a), Point-NN Zhang et al. (2023c) and PointMLP Ma et al. (2022), for the 3D shape classification task on the synthetic ModelNet40 Wu et al. (2015) benchmark. Following the standard practice in the community, we report the class-average accuracy (mAcc) and overall accuracy (OA) on the testing set in Table 1.

When the embedding dimension is set to 36, our PointNet outperforms PointMLP by 0.6% in OA (93.9% vs. 93.3%) and 0.7% in mAcc (91.5% vs. 90.8%), all while requiring 12 times fewer parameters (1.1 M vs. 12.6 M) and 7 times fewer FLOPs (2.0 G vs. 14.6 G). Moreover, compared to PointMLP, our method proves to be more cost-effective in terms of training times, cutting it in half (2.4 hours vs. 6.2 hours). Compared to other methods, including 3DCTN and PointNeXt, our method achieves better performance with fewer parameters. These results validate the efficiency and superiority of our method in shape classification tasks. Additionally, when the embedding dimension is increased to 66, the performance of PointNet does not change. Therefore, in subsequent ModelNet40 experiments, we set the embedding dimension to 36.

5.3. Shape classification on ScanObjectNN

Although our PointNet has already demonstrated its efficiency in shape classification tasks on the synthetic ModelNet40, to thoroughly validate this capability, we extend our evaluation to the real-world ScanObjectNN Uy et al. (2019), where we compare PointNet with several recent methods, including PointNet Qi et al. (2017a), PointNet++ Qi et al. (2017b), PointCNN Li et al. (2018), SpiderCNN Xu et al. (2018), SimpleView Goyal et al. (2021), GBNet Qiu et al. (2022), 3DCTN Lu et al. (2022), PointNeXt Qian et al. (2022), SN-Adapter Zhang et al. (2023b), PointGL Li et al. (2024), PointWavelet Wen et al. (2023), PointMLP Ma et al. (2022) and Point-NN Zhang et al. (2023c).

As shown in Table 2, when the embedding dimension is set to 36, our PointNet utilizes only one-fourteenth of the parameters used by PointMLP (0.9 M vs. 12.6 M), yet achieves superior performance. Specifically, our method surpasses PointMLP by 1.3% in

Table 2
Shape classification on the real-world ScanObjectNN. We report the class-average accuracy (mAcc) and overall accuracy (OA) on the testing set. The best performance value is in **bold**, and the second-best is underlined. Note that C denotes the embedding dimension of PointeNet and $*$ means the reproduced results.

Method	mAcc (%)	OA (%)	Param.	FLOPs	Train Time
PointNet	63.4	68.2	3.5 M	-	-
PointNet++	75.4	77.9	1.7 M	-	-
PointCNN	75.1	78.5	15.2 M	-	-
SpiderCNN	69.8	73.7	-	-	-
SimpleView	-	80.5	-	-	-
GBNet	77.8	80.5	8.4 M	-	-
3DCTN (Multi-scale)	79.5	81.5	4.2 M	3.7 G	-
PointNeXt-S	85.8	87.7	1.4 M	1.6 G	-
SN-Adapter-PointMLP	84.6	86.3	-	-	-
PointGL	85.2	86.9	-	-	-
PointWavelet-L	85.8	87.7	-	-	-
PointMLP*	83.2	85.0	12.6 M	14.6 G	4.8 h
Point-NN*	-	64.8	0 M	0 G	0 h
PointeNet (C = 36) (Ours)	84.5	86.1	0.9 M	2.1 G	3.1 h
PointeNet (C = 66) (Ours)	86.1	87.6	2.6 M	7.1 G	5.6 h
PointeNet (C = 90) (Ours)	**86.4**	**87.9**	4.5 M	13.2 G	7.8 h

Table 3
Part segmentation on ShapeNetPart. We report mean IoU scores across classes (Cls. mIoU) and instances (Inst. mIoU) on the testing set. The best performance value is in **bold**, and the second-best is underlined. Note that C denotes the embedding dimension of PointeNet and $*$ means the reproduced results.

Method	Cls. mIoU (%)	Inst. mIoU (%)	Param.	FLOPs	Train Time
PointNet	80.4	83.7	8.3 M	-	-
PointNet++	81.9	85.1	1.8 M	-	-
Kd-Net	-	82.3	-	-	-
SpiderCNN	82.4	85.3	-	-	-
SPLATNet	83.7	85.4	-	-	-
CSANet	-	85.7	-	-	-
PointNeXt-S (C = 160)*	81.8	85.5	22.5 M	110.2 G	17 h
FlatNet-P	-	84.9	-	-	-
APES	83.7	**85.8**	-	-	-
G-PointNet++	82.4	85.5	-	-	-
PointMLP*	84.0	85.7	16.0 M	5.8 G	9.3 h
Point-NN*	-	70.7	0 M	0 G	0 h
PointeNet (C = 36) (Ours)	83.7	85.2	5.3 M	1.3 G	3.6 h
PointeNet (C = 66) (Ours)	**84.2**	85.6	8.3 M	2.8 G	5.0 h
PointeNet (C = 90) (Ours)	**84.2**	**85.8**	12.3 M	4.7 G	6.2 h

mAcc (84.5% vs. 83.2%) and 1.1% in OA (86.1% vs. 85.0%). Compared to other methods, including PointNN, 3DCTN, and PointNeXt, our method also maintains a significant advantage. Moreover, our method exhibits a smaller gap between class average accuracy (mAcc) and overall accuracy (OA) compared to PointMLP (1.6% vs. 1.8%), suggesting that PointeNet avoids bias towards any specific category, demonstrating commendable robustness. These results on the real-world ScanObjectNN dataset thoroughly validate the efficiency and superiority of our method in the shape classification task. Additionally, as the embedding dimension is increased to 66 and 90, the performance of PointeNet is consistently improved. Therefore, in the subsequent ScanObjectNN experiments, we set the embedding dimension to 90.

5.4. Part segmentation on ShapeNetPart

We present the results of PointeNet and compare our method with several recent works, including PointNet Qi et al. (2017a), PointNet++ Qi et al. (2017b), Kd-Net Klokov and Lempitsky (2017), SpiderCNN Xu et al. (2018), SPLATNet Su et al. (2018), CSANet Wang et al. (2022), PointNeXt Qian et al. (2022), FlatNet Zhang et al. (2023a), APES Wu et al. (2023), G-PointNet++ Liu and Tian (2024), PointMLP Ma et al. (2022) and Point-NN Zhang et al. (2023c), for the 3D shape part segmentation task on the ShapeNetPart benchmark, as summarized in Table 3.

Analyzing these results, when the embedding dimension is set to 66, our PointeNet achieves comparable segmentation performance to PointMLP with only half the parameters (8.3 M vs. 16.0 M), especially in Cls. mIoU, where it surpasses PointMLP by 0.2% (84.2% vs. 84.0%). And, our method significantly outperforms in training time, requiring only half the time needed for PointMLP (5.0 h vs. 9.3 h). Compared to other methods, including Point-NN, SN-Adapter, and G-PointNet++, our method also demonstrates stronger segmentation capabilities. Furthermore, when trained on a single RTX 4090 GPU, our method shows superior advantages over PointNeXt, without being limited by small batch sizes. Additionally, when the embedding dimension is simply increased to 90,

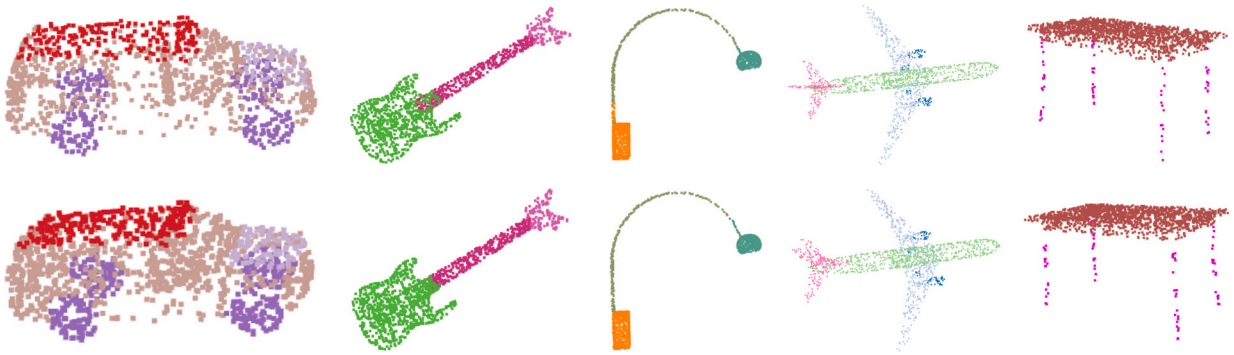

Fig. 4. Visualization of part segmentation results on ShapeNetPart. The top line represents the ground truth, while the bottom line corresponds to our predictions. Upon visualization, our predictions closely align with the ground truth.

Table 4
Results of PointRCNN with and without PointeNet on the KITTI val set. The best performance value is in **bold**, and the second-best is underlined. $*$ means the reproduced results.

Method	Car (3D AP_{R40})			Pedestrian (3D AP_{R40})			Cyclist (3D AP_{R40})		
	Easy	Mod.	Hard	Easy	Mod.	Hard	Easy	Mod.	Hard
SECOND	90.55	81.61	78.56	55.94	51.15	46.17	82.97	66.74	62.78
PointPillars	87.75	78.41	75.19	57.30	51.42	46.87	81.57	62.93	58.98
PV-RCNN	92.10	84.36	82.48	64.26	56.67	51.91	88.88	71.95	66.78
PC-RGNN	90.94	81.43	80.45	-	-	-	-	-	-
Voxel-RCNN	91.72	83.19	78.60	-	-	-	-	-	-
VoxSeT	91.02	82.01	79.04	-	-	-	-	-	-
CAT-Det	90.12	81.46	79.15	-	-	-	-	-	-
VP-Net	89.01	82.19	79.46	-	-	-	-	-	-
FARP-Net	89.91	83.99	79.21	-	-	-	-	-	-
PointRCNN*	91.28	80.78	78.37	65.12	56.43	49.24	90.02	72.42	67.44
PointRCNN + PointeNet	**92.34**	83.13	80.78	**66.00**	**58.62**	**52.09**	**92.99**	**73.97**	**69.50**
Improvement	**+1.06**	**+2.35**	**+2.41**	**+0.88**	**+2.19**	**+2.85**	**+2.97**	**+1.55**	**+2.06**
3DSSD*	91.84	84.52	82.10	59.49	54.69	50.46	89.82	69.73	65.49
3DSSD + PointeNet	92.23	**85.18**	**82.66**	61.76	56.00	51.59	88.82	72.29	67.75
Improvement	**+0.39**	**+0.66**	**+0.56**	**+2.27**	**+1.31**	**+1.13**	-1.00	**+2.56**	**+2.26**

our PointeNet only exhibits a slight improvement, with a 0.2% increase in Cls. mIoU. Therefore, in the subsequent ShapeNetPart experiments, we set the embedding dimension to 66.

Additionally, we present visualizations of the ground truth and predicted segmentation in Fig. 4. Intuitively, the predictions from our PointeNet closely match the ground truth, providing qualitative evidence for the validity of our method. These quantitative and qualitative results all demonstrate the effectiveness and efficiency of our PointeNet.

5.5. 3D object detection on KITTI

To validate the practicality and efficiency of our method in real-world, scene-level applications, we utilize the classic PointRCNN Shi et al. (2019) and 3DSSD Yang et al. (2020) as baselines and evaluate them on the real-world autonomous driving dataset KITTI Geiger et al. (2012). We compare our method with several established 3D object detection networks, including SECOND Yan et al. (2018), PointPillars Lang et al. (2019), PV-RCNN Shi et al. (2020), Voxel-RCNN Deng et al. (2021), PC-RGNN Zhang et al. (2021), VoxSeT He et al. (2022), CAT-Det Zhang et al. (2022), VP-Net Song et al. (2023) and FARP-Net Xie et al. (2024).

The findings from Table 4 reveal that our PointeNet considerably enhances the performance of both PointRCNN and 3DSSD across the categories of *Car*, *Pedestrian*, and *Cyclist*, with the most notable improvements exceeding 2%. Notably, even the somewhat dated PointRCNN, when augmented with our PointeNet, achieves a performance level comparable to PV-RCNN, which benefits from the synergistic use of both Voxel- and Point-based structures. Furthermore, the integration of PointeNet enables 3DSSD to outperform its eleven competitors.

Fig. 5 further highlights the qualitative impact of our PointeNet on PointRCNN and 3DSSD, showing that our method significantly reduces both false detections and missed detections, thereby boosting the overall detection performance. These experimental outcomes conclusively prove that our method not only stands out in real-world application scenarios but also demonstrates exceptional efficiency.

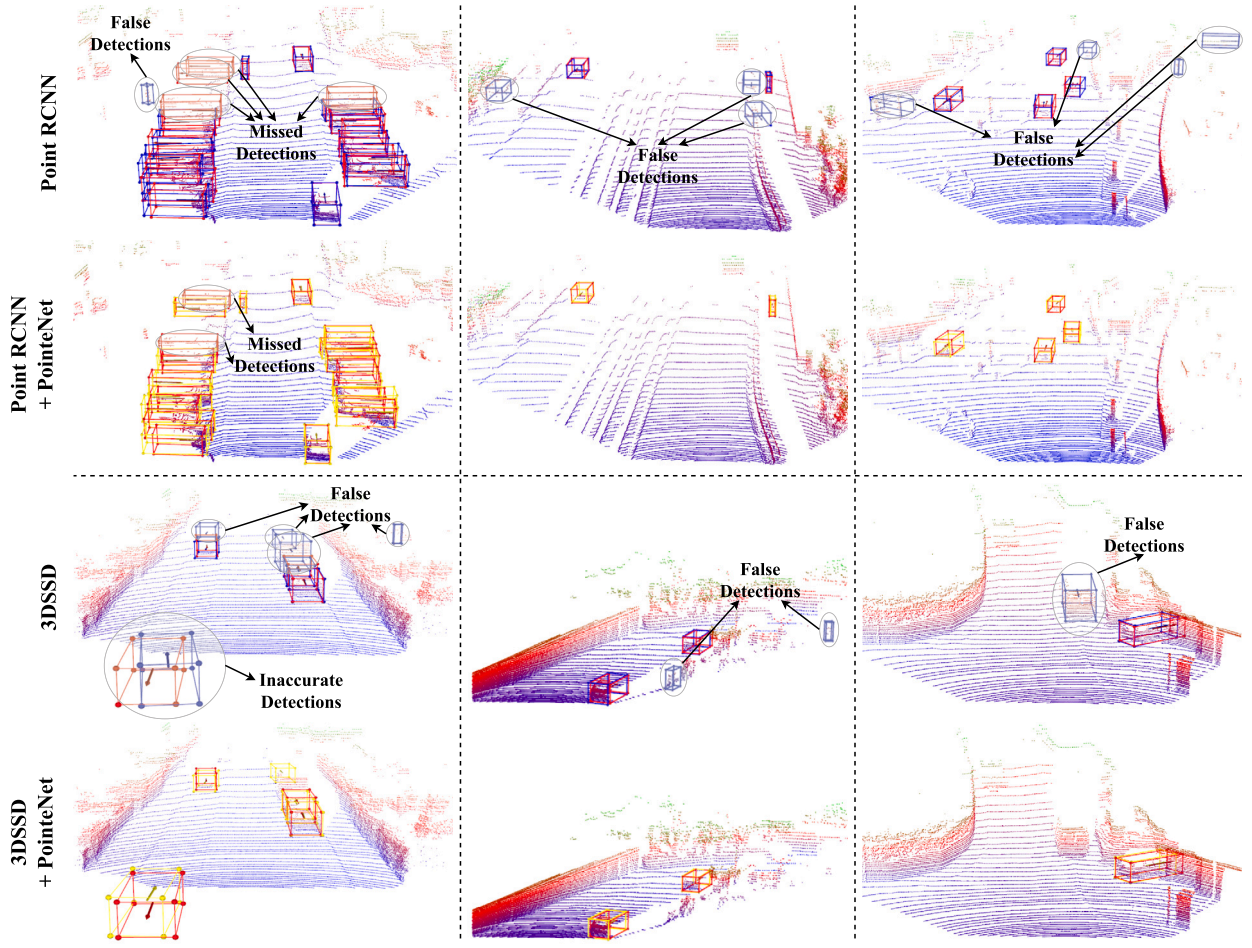

Fig. 5. Visualization of 3D detection results on the KITTI val set. The red 3D bounding boxes represent the ground truth, while the blue/yellow ones denote the prediction results. Upon visualization, our PointeNet aids PointRCNN and 3DSSD in reducing both missed detections and false detections, enhancing the overall detection accuracy. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

Table 5
Ablation results of the proposed MLGA on the ShapeNetPart testing set. We analyze the effect of the multivariate geometries, including spatial neighboring, normal, and curvature, as well as the multivariate feature aggregation strategy (i.e., MAA or Pointwise Addition), on PointeNet. Note that the embedding dimension of PointeNet is set to 36.

Geometries			Aggregation		Cls. mIoU (%)	Inst. mIoU (%)	Param.	FLOPs	Train Time
Neighboring	Normal	Curvature	MAA	Add.					
✓					82.98	84.76	4.4 M	1.3 G	3.2 h
✓	✓		✓		83.47	84.94	5.3 M	1.3 G	3.5 h
✓		✓	✓		83.48	85.13	5.3 M	1.3 G	3.5 h
✓	✓	✓		✓	83.65	85.09	4.4 M	1.3 G	3.5 h
✓	✓	✓	✓		**83.73**	**85.17**	5.3 M	1.3 G	3.6 h

5.6. Ablation study

5.6.1. Effects of MLGA

The proposed Multivariate Local Geometric Aggregation (MLGA) module captures multivariate geometric features, including spatial neighboring, curvature, and normal information, and adaptively aggregates these geometric features for enhanced representation. To thoroughly validate the effectiveness of these geometric features and the proposed Multivariate Adaptive Aggregation (MAA) module, we conduct comprehensive experiments, with findings detailed in Table 5. Analysis of the initial three rows reveals that spatial neighboring, curvature, and normal information all make significant contributions to PointeNet. The comparative results of the fourth and fifth rows also show that adaptively aggregating these geometric features through MAA is more effective than simply adding them together pointwise.

Table 6
Ablation results of the proposed DSE on the KITTI val set. We analyze the effect of MGE and DSE (with/without the distance factor d) on PointeNet.

PointRCNN	MGE	DSE		Car (3D AP_{R40})		
		w/o d	w/ d	Easy	Mod.	Hard
✓				91.28	80.78	78.37
✓	✓			91.98	82.35	80.39
✓	✓	✓		92.04	82.51	80.49
✓	✓		✓	**92.34**	**83.13**	**80.78**

Table 7
The effect of noise on PointeNet. We analyze the robustness of PointeNet by introducing different levels of Gaussian noise into the datasets. σ denotes the standard deviation of the introduced Gaussian noise, specified relative to the radius r of the bounding sphere of a point cloud.

Method	ShapeNetPart (C = 66)		ModelNet40 (C = 36)		ScanObjectNN (C = 90)	
	Cls. mIoU (%)	Inst. mIoU (%)	mAcc (%)	OA (%)	mAcc (%)	OA (%)
$\sigma = 0$	84.2	85.6	91.5	93.9	86.4	87.9
$\sigma = 0.1\% * r$	84.4 (+0.2)	85.8 (+0.2)	91.0 (−0.5)	93.2 (−0.7)	86.4 (−0.0)	87.7 (−0.2)
$\sigma = 0.5\% * r$	84.3 (+0.1)	85.6 (−0.0)	90.4 (−1.1)	93.1 (−0.8)	85.6 (−0.8)	86.9 (−1.0)
$\sigma = 1\% * r$	84.1 (−0.1)	85.3 (−0.3)	90.7 (−0.8)	92.9 (−1.0)	84.8 (−1.6)	86.7 (−1.2)
$\sigma = 3\% * r$	82.0 (−2.2)	84.0 (−1.6)	89.1 (−2.4)	91.9 (−2.0)	82.4 (−4.0)	84.2 (−3.7)
$\sigma = 5\% * r$	79.9 (−4.3)	82.3 (−3.3)	87.9 (−3.6)	91.1 (−2.8)	81.0 (−5.4)	83.2 (−4.7)

5.6.2. Effects of DSE

We perform ablation studies on the challenging *Car* category within the KITTI dataset to evaluate the impact of MGE and Distance-aware Semantic Enhancement (DSE), both with and without the distance factor d, on the performance of PointeNet. According to the findings presented in Table 6, the multivariate geometric features produced by MGE contribute significantly to the improved performance of PointRCNN, notably achieving an enhancement of 2.02% on the hard level. Absent the distance factor to adaptively modulate the segmentation difficulty for semantic features, a slight improvement of 0.1% is observed across all three levels of difficulty. However, when DSE incorporates the distance factor, PointeNet further boosts the performance of PointRCNN, particularly marking a performance increase of 0.78% on the moderate level.

5.6.3. Effects of noise perturbation

The superior performance of PointeNet can largely be attributed to its ability to capture multivariate geometric features, particularly local surface normals and curvature. However, such geometric features are normally considered sensitive to noise. To comprehensively evaluate the robustness of PointeNet against noise, we introduce different levels of Gaussian noise into the datasets and test our method on both part segmentation and classification tasks.

These results are detailed in Table 7. It can be observed that introducing noise to the input point clouds adversely impacts the performance of PointeNet across all evaluated datasets. It is worth noting that when Gaussian noise is at lower levels (σ at 0.1% and 0.5% of the bounding sphere's radius of a point cloud), the performance of the part segmentation task on ShapeNetPart has slightly increased, while that of the classification task on ModelNet40 and ScanObjectNN has slightly decreased. This could be explained by the fact that ShapeNetPart is the most challenging dataset, and adding lower levels of noise to the point clouds instead serves to enhance the sample data to some extent, thereby facilitating a slight performance improvement. Conversely, ModelNet40 and ScanObjectNN are relatively simpler datasets, even slight noise leads to a decline in performance. The most significant decline in performance is observed on ScanObjectNN. This is because ScanObjectNN is collected from the real world and inherently contains certain levels of noise. The introduction of Gaussian noise exacerbates the existing noise, resulting in more pronounced performance drops.

These experimental results reveal that our method demonstrates good robustness under lower-level noise conditions, particularly evident on ShapeNetPart. However, it inevitably suffers significant performance degradation under higher-level noise conditions.

6. Conclusion

While prior methods in point cloud analysis have demonstrated promising success, they often face challenges associated with increased complexity, limiting their applicability to real-world scene-level scenarios and posing efficiency concerns. In response to these challenges, this paper introduces a lightweight network specifically optimized for scalability in real-world autonomous driving applications, offering a more efficient approach to point cloud analysis. Our proposed network leverages non-learnable components, such as FPS and K-NN, and introduces a minimal number of learnable parameters through the proposed Multivariate Geometric Encoding (MGE) module. This module adeptly captures and adaptively aggregates multivariate geometric features, ensuring a lightweight yet comprehensive representation of the point cloud. Furthermore, we incorporate a Distance-Aware Semantic Enhancement (DSE) module tailored for autonomous driving scenarios, capturing distance-aware semantic features. The integration of both multivariate geometric and semantic features contributes to a more efficient and superior performance in point cloud analysis. In future work, we aim to explore more potential point cloud features to further enhance the capabilities of our lightweight network.

CRediT authorship contribution statement

Lipeng Gu: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Investigation. **Xuefeng Yan:** Writing – review & editing, Writing – original draft. **Liangliang Nan:** Writing – review & editing, Writing – original draft. **Dingkun Zhu:** Writing – review & editing, Writing – original draft. **Honghua Chen:** Writing – review & editing. **Weiming Wang:** Writing – review & editing. **Mingqiang Wei:** Writing – review & editing, Writing – original draft.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. T2322012, No. 62172218, No. 62032011), the Shenzhen Science and Technology Program (No. JCYJ20220818103401003, No. JCYJ20220530172403007), the Guangdong Basic and Applied Basic Research Foundation (No. 2022A1515010170), the National Defense Basic Scientific Research Program of China (No. JCKY2020605C003), the Hong Kong Metropolitan University (HKMU) Research Grant (No. RD/2022/2.13), and the HKMU 2023/24 S&T Research Grant (No. R5126).

References

Deng, J., Shi, S., Li, P., Zhou, W., Zhang, Y., Li, H., 2021. Voxel r-cnn: towards high performance voxel-based 3d object detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 1201–1209.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16x16 words: transformers for image recognition at scale. In: International Conference on Learning Representations.

Felzenszwalb, P.F., Huttenlocher, D.P., 2004. Efficient graph-based image segmentation. Int. J. Comput. Vis. 59, 167–181.

Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? The KITTI vision benchmark suite. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3354–3361.

Goyal, A., Law, H., Liu, B., Newell, A., Deng, J., 2021. Revisiting point cloud shape classification with a simple and effective baseline. In: Proceedings of the 38th International Conference on Machine Learning, pp. 3809–3820.

Guo, M., Cai, J., Liu, Z., Mu, T., Martin, R.R., Hu, S., 2021. PCT: point cloud transformer. Comput. Vis. Media 7, 187–199.

He, C., Li, R., Li, S., Zhang, L., 2022. Voxel set transformer: a set-to-set approach to 3d object detection from point clouds. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 8407–8417.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778.

Klokov, R., Lempitsky, V.S., 2017. Escape from cells: deep kd-networks for the recognition of 3d point cloud models. In: IEEE International Conference on Computer Vision, pp. 863–872.

Lai, X., Liu, J., Jiang, L., Wang, L., Zhao, H., Liu, S., Qi, X., Jia, J., 2022. Stratified transformer for 3d point cloud segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 8490–8499.

Lang, A.H., Vora, S., Caesar, H., Zhou, L., Yang, J., Beijbom, O., 2019. Pointpillars: fast encoders for object detection from point clouds. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 12697–12705.

Li, J., Wang, J., Xu, T., 2024. Pointgl: a simple global-local framework for efficient point cloud analysis. arXiv:2401.11650.

Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B., 2018. Pointcnn: convolution on x-transformed points. In: Advances in Neural Information Processing Systems, pp. 828–838.

Liang, T., Xie, H., Yu, K., Xia, Z., Lin, Z., Wang, Y., Tang, T., Wang, B., Tang, Z., 2022. Bevfusion: a simple and robust lidar-camera fusion framework. In: Advances in Neural Information Processing Systems, pp. 10421–10434.

Lin, T., Goyal, P., Girshick, R.B., He, K., Dollár, P., 2020. Focal loss for dense object detection. IEEE Trans. Pattern Anal. Mach. Intell. 42, 318–327.

Lin, Z., Huang, S., Wang, Y.F., 2022. Learning of 3d graph convolution networks for point cloud analysis. IEEE Trans. Pattern Anal. Mach. Intell. 44, 4212–4224.

Liu, H., Tian, S., 2024. Deep 3d point cloud classification and segmentation network based on gatenet. Vis. Comput. 40, 971–981.

Liu, Y., Fan, B., Xiang, S., Pan, C., 2019. Relation-shape convolutional neural network for point cloud analysis. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 8895–8904.

Lu, D., Xie, Q., Gao, K., Xu, L., Li, J., 2022. 3dctn: 3d convolution-transformer network for point cloud classification. IEEE Trans. Intell. Transp. Syst. 23, 24854–24865.

Ma, X., Qin, C., You, H., Ran, H., Fu, Y., 2022. Rethinking network design and local geometry in point cloud: a simple residual MLP framework. In: The Tenth International Conference on Learning Representations.

Qi, C.R., Litany, O., He, K., Guibas, L.J., 2019. Deep Hough voting for 3d object detection in point clouds. In: IEEE International Conference on Computer Vision, pp. 9276–9285.

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017a. Pointnet: deep learning on point sets for 3d classification and segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 77–85.

Qi, C.R., Yi, L., Su, H., Guibas, L.J., 2017b. Pointnet++: deep hierarchical feature learning on point sets in a metric space. In: Advances in Neural Information Processing Systems, pp. 5099–5108.

Qian, G., Li, Y., Peng, H., Mai, J., Hammoud, H., Elhoseiny, M., Ghanem, B., 2022. Pointnext: revisiting pointnet++ with improved training and scaling strategies. In: Advances in Neural Information Processing Systems.

Qiu, S., Anwar, S., Barnes, N., 2022. Geometric back-projection network for point cloud classification. IEEE Trans. Multimed. 24, 1943–1955.

Ran, H., Zhuo, W., Liu, J., Lu, L., 2021. Learning inner-group relations on point clouds. In: IEEE International Conference on Computer Vision, pp. 15457–15467.

Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., Li, H., 2020. PV-RCNN: point-voxel feature set abstraction for 3d object detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 10526–10535.
Shi, S., Wang, X., Li, H., 2019. Pointrcnn: 3d object proposal generation and detection from point cloud. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–779.
Song, Z., Wei, H., Jia, C., Xia, Y., Li, X., Zhang, C., 2023. Vp-net: voxels as points for 3-d object detection. IEEE Trans. Geosci. Remote Sens. 61, 1–12.
Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M., Kautz, J., 2018. Splatnet: sparse lattice networks for point cloud processing. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2530–2539.
Thomas, H., Qi, C.R., Deschaud, J., Marcotegui, B., Goulette, F., Guibas, L.J., 2019. Kpconv: flexible and deformable convolution for point clouds. In: IEEE International Conference on Computer Vision, pp. 6410–6419.
Tian, X., Ran, H., Wang, Y., Zhao, H., 2023. Geomae: masked geometric target prediction for self-supervised point cloud pre-training. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 13570–13580.
Uy, M.A., Pham, Q., Hua, B., Nguyen, D.T., Yeung, S., 2019. Revisiting point cloud classification: a new benchmark dataset and classification model on real-world data. In: IEEE International Conference on Computer Vision, pp. 1588–1597.
Wang, G., Zhai, Q., Liu, H., 2022. Cross self-attention network for 3d point cloud. Knowl.-Based Syst. 247, 108769.
Wang, X., Girshick, R.B., Gupta, A., He, K., 2018. Non-local neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 7794–7803.
Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M., 2019. Dynamic graph CNN for learning on point clouds. ACM Trans. Graph. 38, 146:1–146:12.
Wen, C., Long, J., Yu, B., Tao, D., 2023. Pointwavelet: learning in spectral domain for 3d point cloud analysis. arXiv:2302.05201.
Wu, C., Zheng, J., Pfrommer, J., Beyerer, J., 2023. Attention-based point cloud edge sampling. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 5333–5343.
Wu, W., Qi, Z., Li, F., 2019. Pointconv: deep convolutional networks on 3d point clouds. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 9621–9630.
Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J., 2015. 3d shapenets: a deep representation for volumetric shapes. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1912–1920.
Xie, T., Wang, L., Wang, K., Li, R., Zhang, X., Zhang, H., Yang, L., Liu, H., Li, J., 2024. Farp-net: local-global feature aggregation and relation-aware proposals for 3d object detection. IEEE Trans. Multimed. 26, 1027–1040.
Xu, M., Ding, R., Zhao, H., Qi, X., 2021. Paconv: position adaptive convolution with dynamic kernel assembling on point clouds. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3173–3182.
Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y., 2018. Spidercnn: deep learning on point sets with parameterized convolutional filters. In: Computer Vision - ECCV 2018 - 15th European Conference, pp. 90–105.
Yan, Y., Mao, Y., Li, B., 2018. Second: sparsely embedded convolutional detection. Sensors 18, 3337.
Yang, Z., Jiang, L., Sun, Y., Schiele, B., Jia, J., 2022. A unified query-based paradigm for point cloud understanding. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 8531–8541.
Yang, Z., Sun, Y., Liu, S., Jia, J., 2020. 3dssd: point-based 3d single stage object detector. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 11037–11045.
Yang, Z., Sun, Y., Liu, S., Shen, X., Jia, J., 2019. STD: sparse-to-dense 3d object detector for point cloud. In: IEEE International Conference on Computer Vision, pp. 1951–1960.
Yi, L., Kim, V.G., Ceylan, D., Shen, I., Yan, M., Su, H., Lu, C., Huang, Q., Sheffer, A., Guibas, L.J., 2016. A scalable active framework for region annotation in 3d shape collections. ACM Trans. Graph. 35, 210:1–210:12.
Zhang, Q., Hou, J., Qian, Y., Zeng, Y., Zhang, J., He, Y., 2023a. Flattening-net: deep regular 2d representation for 3d point cloud analysis. IEEE Trans. Pattern Anal. Mach. Intell. 45, 9726–9742.
Zhang, R., Wang, L., Guo, Z., Shi, J., 2023b. Nearest neighbors meet deep neural networks for point cloud analysis. In: IEEE Winter Conference on Applications of Computer Vision, pp. 1246–1255.
Zhang, R., Wang, L., Guo, Z., Wang, Y., Gao, P., Li, H., Shi, J., 2023c. Parameter is not all you need: starting from non-parametric networks for 3d point cloud analysis. arXiv:2303.08134.
Zhang, W., Zhou, H., Sun, S., Wang, Z., Shi, J., Loy, C.C., 2019. Robust multi-modality multi-object tracking. In: IEEE International Conference on Computer Vision, pp. 2365–2374.
Zhang, Y., Chen, J., Huang, D., 2022. Cat-det: contrastively augmented transformer for multi-modal 3d object detection. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 908–917.
Zhang, Y., Huang, D., Wang, Y., 2021. Pc-rgnn: point cloud completion and graph neural network for 3d object detection. In: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 3430–3437.
Zhao, H., Jiang, L., Jia, J., Torr, P.H.S., Koltun, V., 2021. Point transformer. In: IEEE International Conference on Computer Vision, pp. 16239–16248.