WEP Warwick Evans Publishing

International Journal of Computer Science and Information Technology

ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 2, Number 1, Year 2024
DOI: https://doi.org/10.62051/ijcsit.v2n1.19
Journal homepage: https://wepub.org/index.php/IJCSIT/index

Large-scale Point Cloud Segmentation based on Multi-feature Local Enhanced Fusion

Zongshun Wang*

College of Lanzhou University of Technolog, Lanzhou, China.
*Corresponding Author: Zongshun_W@163.com

ABSTRACT

This paper introduces a framework for large-scale 3D point cloud semantic segmentation - the MLEF-Net model. The model aims to improve the segmentation accuracy of large-scale point clouds by innovatively combining Manhattan distance-based KNN neighborhood search with feature aggregation techniques. This approach uniquely handles spatial, color, and normal vector attributes, thereby improving the segmentation results. The superiority of the model is validated through comprehensive testing on the SemanticKITTI and nuScenes datasets, demonstrating its potential to enhance point cloud segmentation through advanced feature fusion strategies.

KEYWORDS

3D scene understanding; Point cloud segmentation; Feature extraction; Manhattan distance

1. INTRODUCTION

With the rapid development of fields such as autonomous driving, robot navigation, and 3D urban modeling, the processing and analysis of point cloud data are becoming increasingly important. Point cloud data, typically collected by devices such as LiDAR scanners, can provide detailed three-dimensional information about the physical world. However, effective processing of point cloud data, especially semantic segmentation in large-scale scenes, remains a challenge, primarily due to its unstructured nature and the enormous volume of data.

Current point cloud segmentation methods[1, 2] mostly focus on improving network architectures or optimizing feature extraction processes to enhance segmentation accuracy and efficiency. However, these methods often overlook the intrinsic multi-feature attributes of point cloud data, such as color, normal vectors, etc., which are crucial for understanding subtle structures in complex scenes. Additionally, traditional feature fusion methods often lead to information confusion and decrease segmentation accuracy when dealing with large-scale point cloud data.

Addressing the shortcomings of existing methods in feature extraction and information confusion, we propose a novel point cloud semantic segmentation framework - MLEF-Net, which adopts a multi-feature local enhancement fusion strategy to effectively improve feature representation and segmentation accuracy. Our approach first performs neighborhood search on the point cloud using a Manhattan distance-based KNN algorithm to better capture local geometric structures. Subsequently, the feature individual aggregation module independently processes various features such as spatial positions, colors, and normal vectors, avoiding the information confusion issue in traditional methods. Furthermore, our network structure incorporates the latest advances in deep learning, such as skip connections and progressive feature fusion techniques, to enhance the model's learning capability and preserve detailed information. Finally, we validate the effectiveness of our method through

experiments on the SemanticKITTI and nuScenes datasets, comparing it with existing state-of-the-art methods.

The innovations of this paper are summarized as follows:

- We propose a Manhattan distance-based KNN local neighborhood search algorithm, which reduces computational complexity and improves the efficiency of handling large-scale point cloud data compared to traditional methods using Euclidean distance.
- We introduce a feature individual local enhancement aggregation module, which effectively avoids information confusion and enhances the model's ability to recognize diverse surface features in complex urban environments by independently processing the three-dimensional position, color, and normal vector features of each point.

2. RELATED WORK

3D Point Cloud Processing is an important research direction in the field of computer vision, mainly involving the analysis and processing of point cloud data in three-dimensional space, with the segmentation module being used to achieve tasks such as object recognition, scene understanding, and map generation. In recent years, with the development of deep learning technology, methods for three-dimensional point cloud processing based on deep neural networks have made significant improvements in accuracy and efficiency. Currently, deep learning-based methods for point cloud processing are mainly divided into projection-based, voxel-based, and point-based methods.

2.1. Projection-Based Semantic Segmentation of 3D Point Clouds

These methods primarily project three-dimensional point cloud data onto a two-dimensional plane, transforming the problem into a two-dimensional image processing task, and then utilize mature two-dimensional image semantic segmentation techniques for processing. The key to this method lies in effectively encoding three-dimensional information into two-dimensional images and addressing the information loss caused by projection.

MV3D[3] proposed a method for projecting point clouds from multiple viewpoints and fusing them for 3D object detection. Although primarily used for object detection, the idea of multi-view fusion is also inspiring for semantic segmentation. SqueezeSeg[4] focuses on real-time road object segmentation from LiDAR point clouds by projecting the point cloud into two-dimensional images and using CNNs for segmentation, demonstrating the effectiveness of transforming 3D point clouds into 2D images for processing. RangeNet++[5] further improves semantic segmentation from point clouds to 2D images by using more efficient neural networks and post-processing techniques to improve segmentation accuracy and speed.

2.2. Voxel-Based Semantic Segmentation of 3D Point Clouds

This method discretizes point cloud data into voxel grids and then uses methods similar to processing three-dimensional images for segmentation. The key challenge here is how to effectively manage memory and computational resources while maintaining high resolution.

VoxNet[6] is one of the early studies to use 3D convolutional neural networks for real-time object recognition from voxelized point cloud data, providing a foundation for subsequent research on point cloud semantic segmentation. 3D R-CNN[7] reconstructs instance-level 3D objects by converting point cloud data into voxel representations and combining rendering and comparison mechanisms, demonstrating the potential of voxel methods in handling complex 3D structures. SparseConvNet[8] introduces an efficient sparse convolution operation for handling high-resolution 3D voxel data, significantly improving computational and storage efficiency.

2.3. Point-Based Semantic Segmentation of 3D Point Clouds

The essence of point clouds is an unordered set of points, and point-based methods directly operate on these points, avoiding information loss during discretization.

PointNet [9]is the first deep learning architecture to operate directly on point clouds, handling the unorderedness of points through a symmetric function and learning global features, laying the foundation for subsequent research on point cloud processing. PointNet++[10] introduces a hierarchical learning mechanism based on PointNet, which can better capture the local structural features of point cloud data. DGCNN[11] effectively learns the local neighborhood structure in point clouds through dynamic graph convolutional networks, improving the performance of classification and segmentation tasks. However, point-based methods are often affected by FPS and KNN algorithms, resulting in very slow model inference speeds. To address this issue, this paper improves segmentation performance under large-scale point clouds by applying a simpler random sampling algorithm and a Manhattan distance-based KNN local search algorithm instead of FPS sampling, and further optimizes the local feature extraction capability of point clouds using a feature individual aggregation module.

3. METHOD

In this paper, we first provide a detailed introduction to the framework of MLEF-Net in Section 3.1. In Section 3.2, we explore the specific process of the KNN neighborhood search algorithm based on Manhattan distance and compare it with the Euclidean distance. In Section 3.3, we elucidate the issues with traditional feature aggregation and further propose a multi-feature local enhancement aggregation module to address the problem of information confusion caused by traditional feature aggregation.

3.1. Framework Overview

The framework diagram of our MLEF-Net network for three-dimensional point cloud segmentation is shown in Figure 1. It adopts the popular U-Net structure, forming the overall structure of the network in an encoder-decoder manner. In the network architecture of MLEF-Net, the system first receives preprocessed N sampled points with D-dimensional features. By applying a series of encoder-decoder layers, the network can capture detailed features of each point. The encoder-decoder process involves five different feature dimensions: 8, 32, 128, 256, 512. This progressive deep network design helps gradually increase the receptive field, thereby better handling complex three-dimensional point clouds in large urban environments. To enhance feature learning, an MLEF module is embedded in each encoding stage, efficiently integrating the normal vectors, colors, and geometric shape information of the point cloud. Additionally, by adopting random sampling at each layer, the network reduces the number of point clouds while aggregating feature information and establishes local regions of point cloud scenes using Manhattan distance, with a sampling rate set to 1/4, aiming to reduce computational load. As the feature increases from the original 32 dimensions to 512 dimensions, the spatial position information of the point cloud may gradually be lost, which is critical in three-dimensional point cloud semantic segmentation in large-scale urban scenes. To address this issue, the network utilizes skip connection technology to fuse low-dimensional and high-dimensional features, preventing information loss due to increased network depth. In the encoder-decoder stage, through four such skip connections, the network ensures that sufficient spatial position information is retained during deep-level processing. Thus, in the final output stage, even with the restoration of feature dimensions and point cloud quantity through multi-layer perceptron (MLP) and nearest neighbor interpolation upsampling methods, the segmentation result accuracy can be ensured.

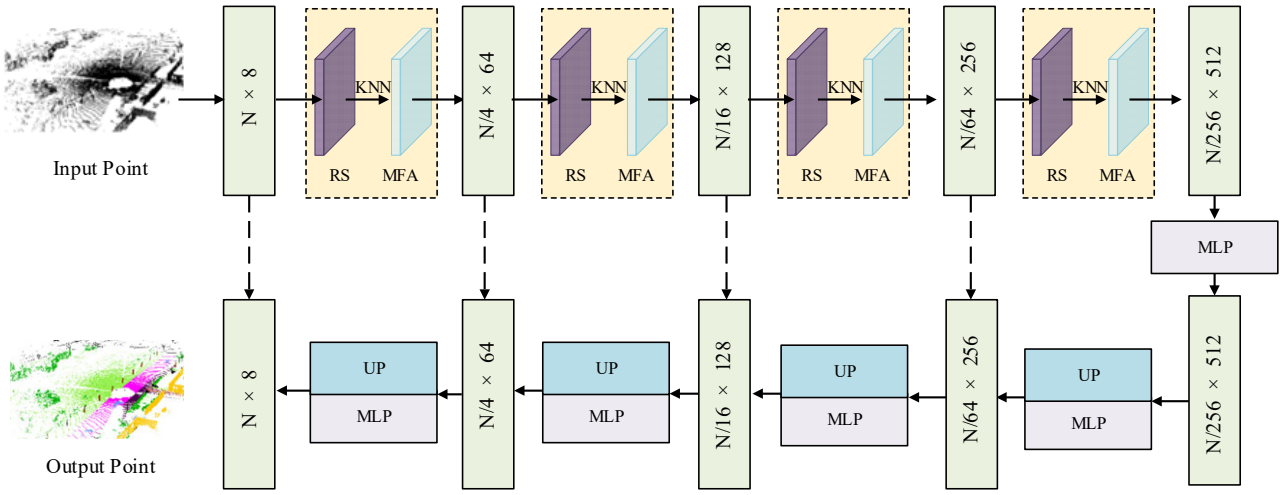

Figure 1. Overall Framework of MLEF-Net.

3.2. Manhattan Distance-based KNN Neighborhood Search Algorithm

In three-dimensional point cloud segmentation algorithms, constructing local neighborhoods is a core step crucial for understanding the local structure and characteristics of point clouds. Different objects and scene features exhibit significant differences in scale. By adjusting the size of the local neighborhood, segmentation algorithms can flexibly adapt to structures of different scales, striking a balance between details and the global context. In this paper, our local neighborhood construction first samples three-dimensional point cloud scenes through random sampling to obtain sampled points, based on which the traditional KNN search algorithm uses Euclidean distance to find the K nearest points with the shortest Euclidean distance to the sampled point in Euclidean space. However, Euclidean distance involves square root and square operations, leading to high computational costs, especially in point cloud semantic understanding in large-scale scenes. Therefore, this paper establishes the local features of point clouds more efficiently by using a Manhattan distance-based KNN local neighborhood search algorithm, which is more efficient and suitable for large-scale scenes. This approach not only reduces the computational complexity of the network but also mitigates the negative impact of the "curse of dimensionality" brought by Euclidean distance in high-dimensional data scenes. Through carefully designed local neighborhood construction strategies and optimization techniques, the local geometric information of point clouds can be extracted more efficiently.

The goal of K-nearest neighbor (KNN) search is to find the nearest K neighbors for each point in the dataset. In three-dimensional point clouds, each point is represented by its spatial coordinates. The Manhattan distance-based KNN algorithm measures the proximity between points using the Manhattan distance. Manhattan distance, also known as city block distance or L_1 norm, is a concept in geometry. Unlike Euclidean distance, Manhattan distance can only move along the directions of the coordinate axes when calculating the distance between two points. First, select the value of K, which represents the number of neighbors desired within the neighborhood of each point, and define an empty neighbor list for each point in the point cloud dataset. For each sampled point P_i with coordinates (x_i, y_i, z_i) in the point cloud, compute its Manhattan distance to all other points P_j with coordinates (x_j, y_j, z_j):

$$D_M(P_i, P_j) = |x_i - x_j| + |y_i - y_j| + |z_i - z_j| \tag{1}$$

Computing the Manhattan distance involves absolute value and addition operations. In a three-dimensional point cloud space, the time complexity of calculating the Manhattan distance between

two points is $O(N)$ because subtraction and absolute value operations need to be performed for each dimension.

For the same two points, the calculation formula for Euclidean distance is:

$$D_E(P_i,P_j)=\sqrt{(x_i-x_j)^2+(y_i-y_j)^2+(z_i-z_j)^2} \tag{2}$$

Computing Euclidean distance involves subtraction, square, addition, and square root operations. In three-dimensional point cloud space, the time complexity of calculating the Euclidean distance between two points is also $O(N)$ because subtraction and square operations need to be performed for each dimension, and then the results for all dimensions are summed and square rooted. Although the time complexity is the same as that of Manhattan distance, Euclidean distance has a larger constant factor in actual calculations due to the involvement of square and square root operations. As the point cloud scene gradually increases, the computational efficiency of Euclidean distance is lower than that of Manhattan distance.

3.3. Multi-Feature Local Enhancement and Aggregation Module

By employing the k-nearest neighbor (KNN) search algorithm, the system first identifies the three-dimensional coordinates, colors, and normals of each input sampling point p and its k nearest neighbors. This step is to deeply encode the spatial relationships among the sampling point p and its surrounding k points, thereby establishing their connections. The encoding process involves not only the specific positional coordinates of each point but also the relative positional relationships between point p and its neighboring point p_i as well as the Manhattan distance $||p_i-p_j^k||$ between them. Through this approach, the system can output a comprehensive positional encoding R_i^k, capturing the accurate spatial positioning of point p within its neighborhood group. The positional encoding R_i^k can be represented as:

$$R_i^k=MLP(p_i\oplus p_j^k\oplus(p_i-p_j^k)\oplus||p_i-p_j^k||) \tag{3}$$

where $\oplus$ denotes concatenation operation. The purpose of positional encoding is to utilize the relative positions between two points to define the specific orientation of neighboring points around the central point p, while determining their actual distances by calculating the Euclidean distance. Such encoding allows for an accurate grasp of the three-dimensional spatial positions of points within the entire neighborhood using only one central point, thereby revealing the local geometric structure of the central point p and enabling the network to learn the spatial and geometric structural information of point clouds more accurately.

Next, the focus shifts to how to effectively incorporate the color and normal vector features of point clouds. As shown in Figure 2. If using the conventional multi-feature encoding method, both the color and normal vector information of three-dimensional point clouds are directly encoded together with the relative positional features of point clouds during the encoding process. The resulting feature aggregation is as follows:

$$F_i^k=MLP(p_i\oplus p_i^k\oplus(p_i-p_i^k)\oplus||p_i-p_i^k||\oplus C) \tag{4}$$

where C represents additional features. However, this traditional multi-feature aggregation encoding method encodes the geometric features and additional features of point clouds directly together, leading to information confusion. Especially when introducing the point cloud normal vector feature(x_n,y_n,z_n), which is highly similar to the positional information(x,y,z), it becomes even more difficult to distinguish.

In order to more effectively learn point cloud information, we have adopted an innovative approach (Feature individual aggregation method) to process the three-dimensional position, color, and normal vector features of each point, avoiding potential confusion caused by directly mixing

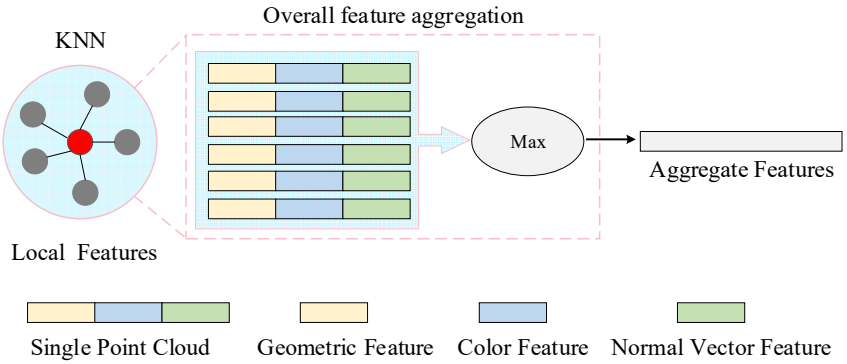

Figure 2. Conventional feature aggregation method

these features. That is, the geometric feature information and additional feature information are encoded separately, and then their encoding results are concatenated together to form a multi-feature joint encoding. This method can better distinguish between different types of features and is more suitable for learning point cloud data.

To consider the separate encoding of multi-feature information, for the color feature, we have adopted a coding method similar to but slightly different from processing geometric position information. Unlike the position encoding, color encoding does not need to consider the absolute differences between features. Specifically, we compute the color information between each point and its k nearest neighbors as well as the color difference between them $c_i - c_i^k$, and then encode them through MLP to form a relative color feature encoding $F_{c_i}^k$ for each point. The overall encoding method is as follows:

$$F_{c_i}^k = MLP\left(c_i \oplus c_i^k \oplus \left(c_i - c_i^k\right)\right) \tag{5}$$

The processing method for normal vector features follows a similar logic. In this process, there is no need to consider the differences in normal vector features. The encoding method for relative normal vector features $F_{n_i}^k$ is as follows:

$$F_{n_i}^k = MLP\left(n_i \oplus n_i^k\right) \tag{6}$$

where n_i represents the normal vector feature of point cloud p_i, and n_i^k represents the normal vector feature of K nearest neighbors.

By finely designing and independently processing the encoding of position, color, and normal vector features, we can maximize the unique value of each feature. Finally, we combine these feature encodings to form a multi-feature joint encoding that includes position, color, and normal vector information, providing a comprehensive feature representation for each point.

As shown in Figure 3. Through the above encoding method, the relative position encoding R_i^k of the k nearest neighbor points and the neighborhood center point, the relative color feature encoding $F_{c_i}^k$, and the relative normal vector feature encoding $F_{n_i}^k$ are concatenated, resulting in the multi-feature joint encoding $\hat{F}_i^k$ of k sampling points, which can be represented as:

$$\hat{F}_i^k = R_i^k \oplus F_{c_i}^k \oplus F_{n_i}^k \tag{7}$$

By constructing multi-feature joint encodings for each point and its neighboring points, we can effectively associate this information with the neighborhood center point while preserving the position, color, and normal vector information of each point. This method ensures that even in the case of a large number of missing data points caused by random sampling during network training, the learning efficiency and accuracy of the network are still guaranteed. More importantly, by incorporating normal vector information, we not only compensate for the limitations of relying solely on position and color information but also significantly enhance the model's recognition ability for different surface textures and undulations in complex urban environments. This in-depth feature understanding, especially in handling objects with diverse surface features in urban scenes, significantly improves the network's recognition and learning capabilities.

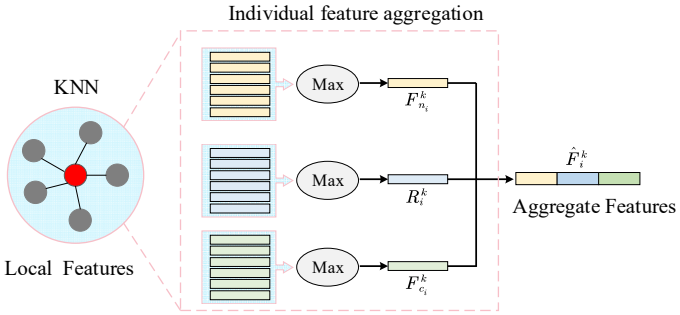

Figure 3. Feature individual aggregation method

4. EXPERIMENTS

In this section, we introduce the datasets used in our experiments in Section 4.1, describe the experimental details from the aspects of parameter design of the network structure and model training strategy in Section 4.2, and further analyze the experimental results of MLEF-Net on the validation dataset in Section 4.3. In Section 4.4, we conduct ablation experiments on the key steps of the model.

4.1. DATASETS

SemanticKITTI. SemanticKITTI is a dataset designed for semantic segmentation of 3D point clouds in autonomous driving, created jointly by the Karlsruhe Institute of Technology in Germany and the Toyota Research Institute. The dataset contains over 23 million point cloud frames, each frame containing approximately 100,000 3D points, mainly collected by high-precision LiDAR scanners mounted on vehicles in diverse environments. The data is preprocessed, including denoising and filtering, to improve quality. Each point is accurately labeled as a semantic category, such as road, vehicle, etc., with a total of 28 categories, ensuring labeling quality through experts and specialized tools. The dataset is divided into training, validation, and test sets to support model training and evaluation, providing valuable resources for the development and application research of point cloud processing technologies in fields such as autonomous driving.

nuScenes. The nuScenes dataset, created by Aptiv, aims to support research on autonomous driving systems, particularly in the field of semantic segmentation of 3D point clouds. The dataset is collected in diverse urban environments in Boston and Singapore, containing 1000 scenes, 140 million point cloud frames, and accompanying panoramic camera images, radar, and IMU data. The data is collected by vehicles equipped with multiple sensors and undergoes strict preprocessing to ensure

synchronization and consistency between sensor data. Objects in each scene are meticulously annotated with detailed 3D bounding boxes covering 23 object categories, ensuring the accuracy and consistency of the data through precise labeling by a professional team. nuScenes explicitly divides the dataset into training, validation, and test sets, providing researchers with a comprehensive and accurate data resource to promote technological development and research progress in the field of autonomous driving.

4.2. Implementation Details and Settings

In this study, we built the experimental environment on the Linux Ubuntu 18.04 operating system, using 4 RTX 3090 GPUs for GPU acceleration with CUDA 11.2. We selected PyTorch 1.9 based on Python 3.6 as the deep learning framework. To balance performance and memory consumption, we set the batch size to 8 and trained the model for 200 epochs using the AdamW optimizer. To further enhance the model's performance, we performed a series of data augmentation operations, including random flipping and dropout. These operations help increase the diversity of the data, thereby improving the model's generalization ability. Additionally, we adopted a cosine annealing strategy to adjust the learning rate. Through this strategy, we can dynamically adjust the learning rate during training to better control the training process of the model and improve its performance.

4.3. Results

4.3.1. Evaluation on SemanticKITTI Dataset

The experimental results of our method on the SemanticKITTI dataset are shown in Table 1. MLEF-Net achieves a 5.9% improvement in mIOU evaluation metric compared to the classic point cloud segmentation network AMVNet, and a 1.5% improvement over the AF2S3 model. It ranks higher on most of the 16 compared categories. This result demonstrates that MLEF-Net, by using the MLEF module and progressive deep network design, can effectively integrate normal vector, color, and geometric shape information of point clouds, improving the feature learning capability. The use of skip connection technology may help retain sufficient spatial position information in deep processing, thereby improving segmentation accuracy.To visually verify the performance of MLEF-Net on the SemanticKITTI dataset, we further experiment with visualization results in Figure 4. Unlike other works, we also provide visualization graphs of segmentation errors compared to Ground Truth. The visualization results show that our model performs well in segmenting important point cloud scenes.

Table 1. shows the experimental results on the SemanticKITTI dataset. In the table, we have highlighted the highest mIOU (mean Intersection over Union) scores for each category in red and the second highest scores in blue.

Methods	mIoU(%)	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk
PointNet[9]	14.6	46.3	1.3	0.3	0.1	0.8	0.2	0.2	0.0	61.6	15.8	35.7	1.4	41.4	12.9	31.0	4.6
RandLANet[12]	53.9	94.2	26.0	25.8	40.1	38.9	49.2	48.2	7.2	90.2	60.3	73.7	20.4	86.9	56.3	81.4	61.3
KPConv [13]	58.8	96.0	30.2	42.5	33.4	44.3	61.5	61.6	11.8	88.8	61.3	72.7	31.6	90.5	64.2	84.8	69.2
Squeezev3[14]	55.9	92.5	38.7	36.5	29.6	33.0	45.6	46.2	20.1	91.7	63.4	74.8	26.4	89.0	59.4	82.0	58.7
RangeNet++[5]	52.2	91.4	25.7	34.4	25.7	23.0	38.3	38.8	4.8	91.8	65.0	75.2	27.8	87.4	58.6	80.5	55.1
FusionNet[15]	61.3	95.3	47.5	37.7	41.8	34.5	59.5	56.8	11.9	91.8	68.8	77.1	30.8	92.5	69.4	84.5	69.8
TornadoNet[16]	63.1	94.2	55.7	48.1	40.0	38.2	63.6	60.1	34.9	89.7	66.3	74.5	28.7	91.3	65.6	85.6	67.0
AMVNet[17]	65.3	96.2	59.9	54.2	48.8	45.7	71.0	65.7	11.0	90.1	71.0	75.8	32.4	91.4	69.1	85.6	67.0
SPVCNN[18]	63.8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
SPVNAS[19]	67.0	97.2	50.6	50.4	56.6	58.0	67.4	67.1	50.3	90.2	67.6	75.4	21.8	91.6	66.9	86.1	73.4
AF2S3[20]	69.7	94.5	65.4	86.8	39.2	41.1	80.7	80.4	74.3	91.3	68.8	72.5	53.5	87.9	63.2	70.2	68.5
Ours	71.2	98.6	66.5	80.7	53.2	45.6	72.5	75.4	73.2	92.3	72.5	73.4	55.3	92.7	68.8	74.1	67.4

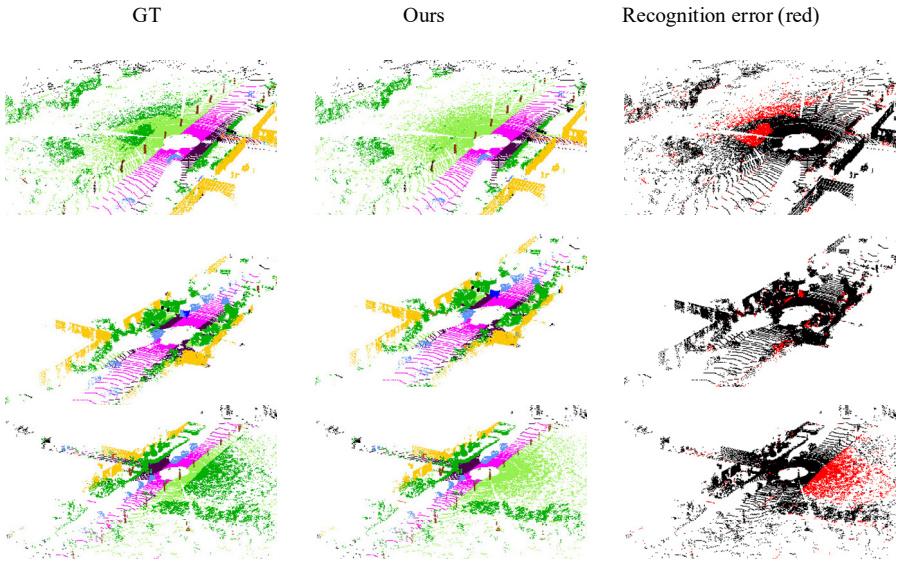

Figure 4. presents the semantic segmentation visualization results of the MLEF-Net network on the SemanticKITTI Dataset. The rightmost column represents the point clouds where prediction errors occurred, which are highlighted in red.

4.3.2. Evaluation on the nuScenes Dataset

To comprehensively validate the robustness of MLEF-Net across various point cloud scenes, we conducted a series of detailed experiments on the nuScenes dataset. According to the quantitative data in Table 2, our model performed excellently in terms of mIOU on the nuScenes dataset, surpassing the AMVNet model by 2.7%. Surprisingly, it outperformed the indicator of the 3D point

cloud semantic segmentation model based on point, voxel, and distance map fusion from multiple perspectives by 1.2%. This is attributed to the feature individual aggregation method employed in our network, which handles the three-dimensional position, color, and normal vector features of each point separately, avoiding the potential confusion caused by directly mixing these features and thereby improving the accuracy of the model in the task of three-dimensional point cloud semantic segmentation.

Table 2. presents the experimental data of MLEF-Net on the nuScenes dataset. We highlighted the highest results in red and the second highest results in blue.

Methods	mIoU（%）	Barrier	Bicycle	Bus	Car	Construction	Motorcycle	Pedestrian	Traffic-cone	Trailer	Truck	Driveable	Other_flat	Sidewalk	Terrain	Manmade	vegetation
RangeNet++[5]	65.5	66.0	21.3	77.2	80.9	30.2	66.8	69.6	52.1	54.2	72.3	94.1	66.6	63.5	70.1	83.1	79.8
PolarNet[21]	71.0	74.7	28.2	85.3	90.9	35.1	77.5	71.3	58.8	57.4	76.1	96.5	71.1	74.7	74.0	87.3	85.7
Salsanext[22]	72.2	74.8	34.1	85.9	88.4	42.2	72.4	72.2	63.1	61.3	76.5	96.0	70.8	71.2	71.5	86.7	84.4
AMVNet[17]	76.1	79.8	32.4	82.2	86.4	62.5	81.9	75.3	72.3	83.5	65.1	97.4	67.0	78.8	74.6	90.8	87.4
Cylinder3D[23]	76.1	76.4	40.3	91.2	92.8	51.3	78.0	78.9	64.9	62.1	84.4	96.8	71.6	76.4	75.4	90.5	87.4
RPVNet[24]	77.6	78.2	43.4	92.7	93.2	49.0	85.7	80.5	66.0	66.9	84.0	96.9	73.5	75.9	76.0	90.6	88.9
Ours	78.8	79.8	45.7	94.7	95.6	52.2	84.6	79.6	68.2	69.9	83.5	97.7	74.8	75.8	78.5	91.2	88.5

4.4. Ablation Study

In this section, to clarify the impact of each key component on the network performance, we conducted a series of ablation experiments. In subsection 4.4.1, to demonstrate the efficiency of our model compared to other advanced networks, we further provided a comparison of model parameters and training time with related networks. In 4.4.2, we also conducted ablation experiments comparing different methods of local neighborhood search and feature aggregation. All the aforementioned experiments were implemented on the SemanticKITTI dataset.

4.4.1. Comparison of Model Parameters and Training Time

Table 3. Comparison of model parameters and training times between MLEF-Net and other advanced segmentation networks

	Parameters (M)	Training duration (s/epoch)
PointNet++	1.36	1039.2
SPG	0.34	2530.3
SparseConv	15.8	1812.5
RandLA-Net	1.79	423.6
MLEF-Net（Ours）	1.35	301.3

Table 3 shows the comparison results of model parameters and training time between MLEF-Net and other advanced segmentation networks. The experiments show that MLEF-Net, while achieving better model segmentation accuracy, also exhibits better metrics in terms of model parameters and training time. This is because the other networks mentioned above use Euclidean distance to calculate the distance between two points during the local feature search process, whereas our model uses a KNN local neighborhood search algorithm based on Manhattan distance, further improving the model's speed.

4.4.2. Comparison of KNN and Feature Aggregation Methods

Table 4. Comparison of the choice of KNN and feature aggregation methods.

KNN		mIOU (%)	Training duration (s/epoch)	Feature aggregation		mIOU (%)	Training duration (s/epoch)
Euclidean	Manhattan			Regular	Aggregate alone		
√		70.5	758.2	√		69.5	386.5
	√	71.2	401.3		√	71.2	401.3

To study the impact of using Euclidean and Manhattan distances in KNN for local neighborhood search on the accuracy and efficiency of the MLEF-Net network, it can be observed in Table 4 that the Manhattan distance we used has better effects in both accuracy and inference speed. In the choice of network feature aggregation methods, the feature individual aggregation module we designed is more conducive to the performance of the model than traditional feature aggregation modules.

5. CONCLUSION

This paper presents MLEF-Net, a new framework for large-scale point cloud segmentation that utilizes multi-feature local enhancement fusion to address the issue of information confusion in traditional feature aggregation. This method combines KNN neighborhood search based on Manhattan distance with individual feature aggregation, effectively capturing the spatial, color, and normal vector features of point clouds and improving segmentation accuracy. Extensive evaluations on SemanticKITTI and nuScenes datasets, along with a series of ablation experiments on key steps of the model, have shown significant performance improvements due to the innovative network architecture and feature fusion techniques.

REFERENCES

[1] H. Chen, T. Xie, M. Liang, W. Liu, and P. X. Liu, "A local tangent plane distance-based approach to 3D point cloud segmentation via clustering," Pattern Recognition, vol. 137, p. 109307, 2023.

[2] A.-T. Tran, H.-S. Le, S.-H. Lee, and K.-R. Kwon, "Pointct: Point central transformer network for weakly-supervised point cloud semantic segmentation," in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024, pp. 3556-3565.

[3] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2017, pp. 1907-1915.

[4] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud," in 2018 IEEE international conference on robotics and automation (ICRA), 2018: IEEE, pp. 1887-1893.

[5] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "Rangenet++: Fast and accurate lidar semantic segmentation," in 2019 IEEE/RSJ international conference on intelligent robots and systems (IROS), 2019: IEEE, pp. 4213-4220.

[6] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in 2015 IEEE/RSJ international conference on intelligent robots and systems (IROS), 2015: IEEE, pp. 922-928.

[7] Z. Li, F. Wang, and N. Wang, "Lidar r-cnn: An efficient and universal 3d object detector," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 7546-7555.

[8] J. Wang, W. Li, M. Zhang, and J. Chanussot, "Large kernel sparse ConvNet weighted by multi-frequency attention for remote sensing scene understanding," IEEE Transactions on Geoscience and Remote Sensing, vol. 61, pp. 1-12, 2023.

[9] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 652-660.

[10] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," Advances in neural information processing systems, vol. 30, 2017.

[11] A. V. Phan, M. Le Nguyen, Y. L. H. Nguyen, and L. T. Bui, "Dgcnn: A convolutional neural network over large-scale labeled graphs," Neural Networks, vol. 108, pp. 533-543, 2018. https://doi.org/10.1016/j.neucom.2018.09.008

[12] Q. Hu et al., "Randla-net: Efficient semantic segmentation of large-scale point clouds," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 11108-11117.

[13] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 6411-6420.

[14] C. Xu et al., "Squeezesegv3: Spatially-adaptive convolution for efficient point-cloud segmentation," in Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16, 2020: Springer, pp. 1-19.

[15] F. Zhang, J. Fang, B. Wah, and P. Torr, "Deep fusionnet for point cloud semantic segmentation," in Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16, 2020: Springer, pp. 644-663.

[16] M. Gerdzhev, R. Razani, E. Taghavi, and L. Bingbing, "Tornado-net: multiview total variation semantic segmentation with diamond inception module," in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021: IEEE, pp. 9543-9549.

[17] V. E. Liong, T. N. T. Nguyen, S. Widjaja, D. Sharma, and Z. J. Chong, "Amvnet: Assertion-based multi-view fusion network for lidar semantic segmentation," arXiv preprint arXiv:2012.04934, 2020.

[18] M. Axelsson, M. Holmberg, S. Serra, H. Ovren, and M. Tulldahl, "Semantic labeling of lidar point clouds for UAV applications," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 4314-4321.

[19] Z. Liu, H. Tang, S. Zhao, K. Shao, and S. Han, "Pvnas: 3d neural architecture search with point-voxel convolution," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 11, pp. 8552-8568, 2021.

[20] R. Cheng, R. Razani, E. Taghavi, E. Li, and B. Liu, "2-s3net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network," in Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 12547-12556.

[21] Y. Zhang et al., "Polarnet: An improved grid representation for online lidar point clouds semantic segmentation," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 9601-9610.

[22] T. Cortinhal, G. Tzelepis, and E. Erdal Aksoy, "Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds," in Advances in Visual Computing: 15th International Symposium, ISVC 2020, San Diego, CA, USA, October 5–7, 2020, Proceedings, Part II 15, 2020: Springer, pp. 207-222.

[23] H. Zhou et al., "Cylinder3d: An effective 3d framework for driving-scene lidar semantic segmentation," arXiv preprint arXiv:2008.01550, 2020.

[24] J. Xu, R. Zhang, J. Dou, Y. Zhu, J. Sun, and S. Pu, "Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation," in Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 16024-16033.