4D-CS: Exploiting Cluster Prior for 4D Spatio-Temporal LiDAR Semantic Segmentation

Jiexi Zhong, Zhiheng Li, Yubo Cui, and Zheng Fang, *Member, IEEE*

***Abstract*—Semantic segmentation of LiDAR points has significant value for autonomous driving and mobile robot systems. Most approaches explore spatio-temporal information of multi-scan to identify the semantic classes and motion states for each point. However, these methods often overlook the segmentation consistency in space and time, which may result in point clouds within the same object being predicted as different categories. To handle this issue, our core idea is to generate cluster labels across multiple frames that can reflect the complete spatial structure and temporal information of objects. These labels serve as explicit guidance for our dual-branch network, 4D-CS, which integrates point-based and cluster-based branches to enable more consistent segmentation. Specifically, in the point-based branch, we leverage historical knowledge to enrich the current feature through temporal fusion on multiple views. In the cluster-based branch, we propose a new strategy to produce cluster labels of foreground objects and apply them to gather point-wise information to derive cluster features. We then merge neighboring clusters across multiple scans to restore missing features due to occlusion. Finally, in the point-cluster fusion stage, we adaptively fuse the information from the two branches to optimize segmentation results. Extensive experiments confirm the effectiveness of the proposed method, and we achieve state-of-the-art results on the multi-scan semantic and moving object segmentation on SemanticKITTI and nuScenes datasets.**

***Index Terms*—Semantic scene understanding, deep learning methods, clustering.**

I. Introduction

SEMANTIC segmentation of LiDAR points is a crucial task in autonomous driving and robotics, aiming at predicting the semantic categories of each point. It is of great significance for the downstream tasks, including the semantic mapping [1] and long-term autonomous navigation [2].

In recent years, several approaches [3], [4], [5], [6], [7], [8], [9] have attempted semantic segmentation on a single LiDAR frame. However, these one-by-one segmentation algorithms ignore some useful temporal knowledge, especially the distinct and complementary observations of objects from past moments, making it difficult to handle cases with occlusion and sparse points. Moreover, due to separating each frame independently, these methods cannot distinguish the motion state of objects in a LiDAR sequence, leading to a ghost effect during mapping.

To overcome the above limitations, several methods adopt multi-scan LiDAR points to restore the complete appearance of objects [10] or exploit spatio-temporal features to improve scene perception ability [11], [12], [13]. In addition, they explore the potential moving information from a LiDAR sequence to identify the motion states of objects. For example, MemorySeg [13] recurrently updates a memory feature to compensate for information loss caused by occlusion in the current frame. SVQNet [10] aggregates information from adjacent historical points for local feature encoding and selects temporal context to complete invisible geometry, leading to promising results.

However, even when considering such temporal information, the lack of proper consideration for instance-level information sometimes leads to points belonging to a single object being categorized into different semantic classes. Specifically, as illustrated in Fig. 1(a), the segmentation results of large vehicles are prone to truncation because the network typically focuses on point-wise classification while ignoring instance-level comprehension. Then, as exhibited in Fig. 1(b), even though the motion state of objects is predicted accurately at a certain moment,

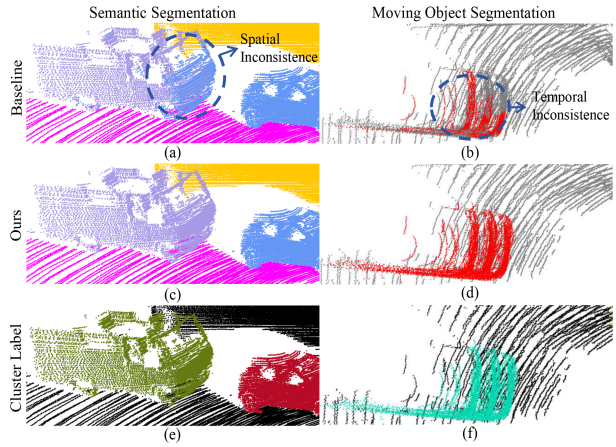

Fig. 1. The comparison of the baseline (WaffleIron [14]) with our proposed method on SemanticKITTI. For both methods, subfigures (a) and (c) display semantic segmentation, while subfigures (b) and (d) illustrate moving object segmentation. Subfigures (e) and (f) present clustering results of foreground objects derived from DBSCAN.

Received 22 July 2024; accepted 14 November 2024. Date of publication 4 December 2024; date of current version 9 December 2024. This article was recommended for publication by Associate Editor Lukas M. Schmid and Editor Cesar Cadena Lerma upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 62073066, in part by the Fundamental Research Funds for the Central Universities under Grant N2226001, and in part by 111 Project under Grant B16009. *(Corresponding author: Zheng Fang.)*

Jiexi Zhong and Zheng Fang are with the Faculty of Robot Science and Engineering, Northeastern University, Shenyang 110819, China, and with the National Frontiers Science Center for Industrial Intelligence and Systems Optimization, Northeastern University, Shenyang 110819, China, and also with the Key Laboratory of Data Analytics and Optimization for Smart Industry, Ministry of Education, Northeastern University, Shenyang 110819, China (e-mail: fangzheng@mail.neu.edu.cn).

Zhiheng Li and Yubo Cui are with the Faculty of Robot Science and Engineering, Northeastern University, Shenyang 110819, China.

The code will be available at https://github.com/NEU-REAL/4D-CS.git.

Digital Object Identifier 10.1109/LRA.2024.3511411

2377-3766 © 2024 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on February 18,2025 at 13:15:05 UTC from IEEE Xplore. Restrictions apply.

it is still difficult for a model to ensure the consistency of segmentation in adjacent time. Thus, how can a model achieve consistent results in both temporal and spatial space? One promising method may be clustering. For outdoor scenes with a sparse distribution of foreground objects, the clustering approaches such as DBSCAN [15] could provide complete object appearance (in Fig. 1(e) and (f)), which is suitable for guiding the network in generating segmentation results that satisfy spatio-temporal consistency.

Building upon this idea, we design a dual-branch segmentation network, called 4D-CS, which views historical features as prior knowledge and further develops cluster-based branch to improve the consistency of segmentation through instance information. Specifically, in the *point-based branch*, we extract point features and adopt a Multi-View Temporal Fusion (MTF) module to enhance them using historical features. Unlike [11], [13], which leverage a memory feature that may accumulate noise to transmit historical knowledge, MTF only considers the most recent historical feature to prevent the sustained influence of incorrect information during inference. Moreover, instead of utilizing a single view in [11], MTF applies past multi-view observations to supplement the spatial features. For the *cluster-based branch*, the intent is to generate cluster labels and utilize them to integrate instance information from point-wise features. Thus, we first employ voxel-based voting to transfer past semantic predictions to the current frame, then use DBSCAN [15] to group foreground objects from multiple frames and aggregate cluster features with pooling. However, the cluster labels do not always fully represent the complete appearance of objects, especially with sparse or occluded point clouds. To address this, we propose a Temporal Cluster Enhancement (TCE) module to collect cluster features from the past frame, improving the integrity of object information. Finally, to strengthen the semantic consistency of points within the same object, we present an Adaptive Prediction Fusion (APF) module in the *point-cluster fusion* stage, which adaptively fuses segmentation results from two branches.

The main contributions of 4D-CS are as follows:

- A dual-branch segmentation network using explicit clustering information to resolve inconsistent predictions of point categories within the same foreground object.
- A novel strategy for obtaining cluster labels, accompanied by three modules: the Multi-view Temporal Fusion, Temporal Cluster Enhancement and Adaptive Prediction Fusion, designed to improve segmentation by utilizing instance information and integrating temporal features.
- The state-of-the-art performance on multi-scan semantic and moving object segmentation on the SemanticKITTI and nuScenes datasets. Our code will be released soon.

II. Related Work

A. Single-Scan Semantic Segmentation

Existing single-scan semantic segmentation algorithms can be classified into point-based, voxel-based, projection-based, and mixture representations. *Point-based algorithms* [3], [4], [5], [16] directly encode features from the raw points. For example, PointNet [5] leverages multi-layer perceptrons (MLP) to extract point-wise features. Then, to improve local structure perception, PointNet++ [4] introduces a hierarchical network for multi-scale information aggregation, while KPConv [3] utilizes kernel-based point convolution to encode local spatial features. However, point-based algorithms are computationally intensive, which constrains their applicability in outdoor scenarios. In contrast, *voxel-based methods* [6], [7] transform unordered points into regular voxels to reduce computational costs. Cylinder3D [6] divides space into cylindrical partitions and avoids redundant processing on empty voxels by sparse convolution. Additionally, SphereFormer [7] presents a radial window to improve the segmentation results of distant points. Yet, these methods are sensitive to voxel size, since big voxel causes information loss while small voxel reduces efficiency. Besides, some *projection-based methods* [8], [9], [17] project point clouds onto 2D planes for feature extraction. PolarNet [17] maps points to polar grids to encode features. CENet [8] and RangeFormer [9] convert LiDAR points to range images. The former adds auxiliary heads for the stronger supervision, while the latter solves many-to-one problem with supervised post-processing. Nevertheless, the projection operation loses much geometric information, limiting segmentation accuracy. Moreover, some *mixture-based methods* [18], [19] attempt to combine the benefits of various representations. RPVNet [19] proposes a range-point-voxel fusion network, integrating the features of different representations by weighted calculation. However, these methods only use a single scan and neglect temporal relationships, resulting in suboptimal segmentation when the LiDAR points are sparse or occluded.

B. Multi-Scan Semantic and Moving Object Segmentation

Since multiple frames contain complete object appearance and reflect the motion state of objects, some works attempt to leverage the spatio-temporal information from multiple scans to enhance the completeness and continuity of segmentation. To reuse valuable history knowledge, MemorySeg [13] maintains a voxel-based memory to improve segmentation results for the current frame. Then, SVQNet [10] completes invisible geometric information at present by searching historical local features surrounding the current points. Moreover, built upon distillation learning, 2DPASS [20] conveys image knowledge to assist with point cloud segmentation. TASeg [21] chooses specific time steps founded on the classification difficulty of each class to stack multiple scans, while also enhancing point features with temporal image features.

Additionally, some methods have focused on using LiDAR sequences to distinguish the motion state of each point [22], [23], [24]. 4DMOS [22] exploits sparse 4D convolution to extract temporal features and applies a Binary Bayes Filter to merge prediction results from different time windows. RVMOS [23] and MF-MOS [24] adopt extra semantic features to assist the model in determining whether the objects are moving or not. However, the above methods do not fully consider semantic consistency, which may result in points belonging to the same object having non-uniform categories.

C. Cluster-Based Semantic Segmentation

Currently, several algorithms [25], [26] have integrated the cluster concept into segmentation to get better performance. DCTNet [25] performs clustering based on feature distance, enabling the network to aggregate the semantically homogeneous points. Later, [26] unites clusters with self-supervised learning and processes point features within the same class to generate

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on February 18,2025 at 13:15:05 UTC from IEEE Xplore. Restrictions apply.

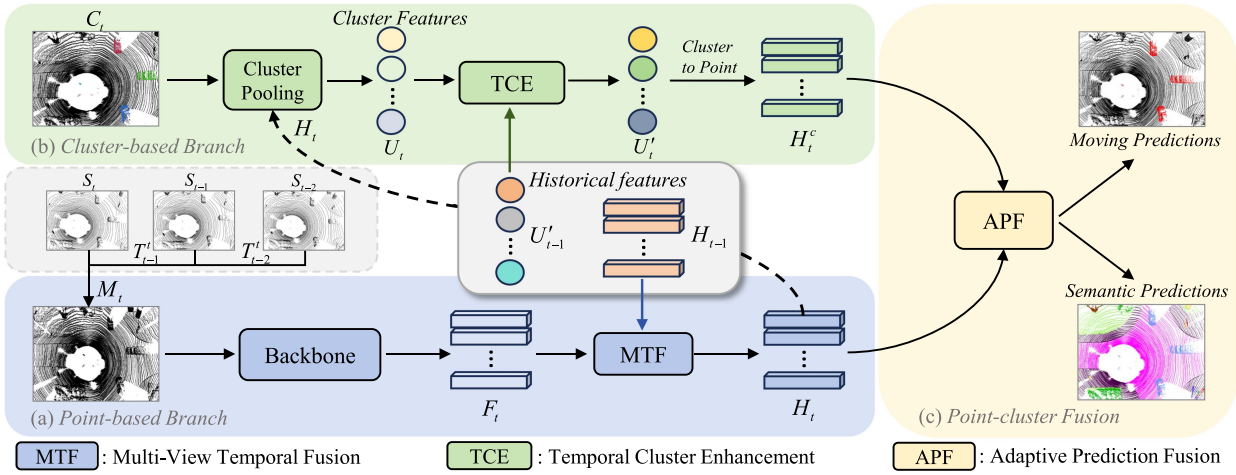

Fig. 2. The framework of our 4D-CS. (a) In the point-based branch, we extract point-wise features and enhance them using historical knowledge through the MTF module. (b) In the cluster-based branch, cluster labels are first used as additional input to generate initial cluster features. The TCE module then integrates adjacent cluster features across multiple frames to enrich instance information, which is subsequently assigned to the corresponding points. (c) Finally, the segmentation results from the two branches are fused adaptively in Point-cluster Fusion.

cluster features. Then, by using contrastive learning, the network could discover latent yet representative subclass patterns. Unlike the above algorithms that perform clustering at the feature level, we explicitly create the cluster labels of foreground objects and guide the network to yield predictions that are consistent in both spatial and temporal dimensions.

III. Methodology

A. Overview

In this section, we propose a cluster-assisted method, 4D-CS, which improves the consistency of segmentation results for points belonging to the same object. As displayed in Fig. 2, our method consists of *point-based branch*, *cluster-based branch*, and *point-cluster fusion*. For the *point-based branch* in Fig. 2(a), we first align multi-scan point clouds using ego-motion and feed them into the backbone network to extract features F_t. To leverage past knowledge, we use the Multi-View Temporal Fusion (MTF) module to merge temporal features on multiple views, resulting in an enhanced feature H_t. For the *cluster-based branch* in Fig. 2(b), we produce the cluster labels C_t based on historical predictions and exploit them to aggregate initial instance features U_t from point features H_t. Later, a Temporal Cluster Enhancement (TCE) module is proposed to integrate temporal cluster features, which are then allocated to foreground points to create refined instance features H_t^c. Finally, for the *point-cluster fusion* in Fig. 2(c), we adopt features from both branches to predict segmentation results, then adaptively optimize the semantic categories and motion states of each point in the Adaptive Prediction Fusion (APF) module.

B. Point-Based Branch

As illustrated in Fig. 2, we utilize the pose transformation matrix T_{t-n}^t to convert past scans $S_{t-n}(n \in \{1, \ldots, N\})$ into the coordinate system of current points S_t. By stacking them, we can get dense point clouds $M_t = \{p_i\}_{i=1}^K (p_i \in \mathbb{R}^5)$, where each point p_i consists of 3D coordinates (x, y, z), intensity r and distance d from the origin of the LiDAR sensor frame. During the point feature extraction, we adopt WaffleIron [14] as our backbone network, which first combines K-Nearest Neighbors (KNN) with MLP to get coarse local features for each point. Thereafter, the points are mapped onto 2D planes of different views to extract features, avoiding the computational burden caused by directly processing numerous point clouds. Specifically, we project point features along the z-axis onto the x-y plane and exploit 2D convolutions to extract semantic information. Subsequently, we back-project 2D features into point clouds and map them again along the y-axis and x-axis onto other planes. Through repeating the above process, we can achieve efficient feature extraction and generate point-wise features $F_t \in \mathbb{R}^{N_p \times D}$, where N_p is the number of downsampled points.

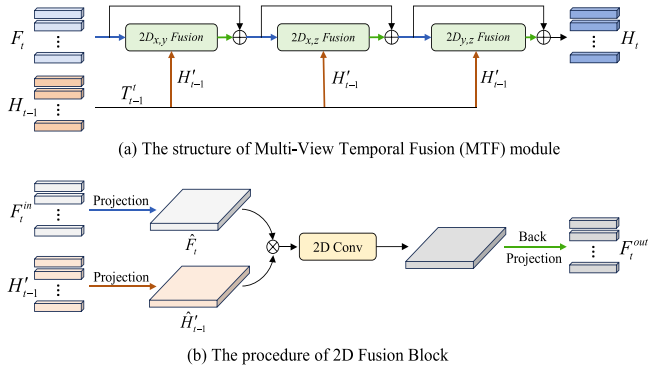

Fig. 3. In the MTF module shown in (a), we sequentially fuse the current and historical features on the x-y, y-z, and x-z planes using the 2D Fusion module illustrated in (b) to integrate the 3D spatial features efficiently.

Multi-View Temporal Fusion: To leverage temporal information fully, we utilize a MTF module to combine historical information with the current features. Initially, the projection matrix T_{t-1}^t is applied to transform historical features H_{t-1} to the current frame's coordinate system. Then, as shown in Fig. 3(a), we sequentially feed the transformed features H'_{t-1} and F_t into 2D fusion blocks corresponding to x-y, x-z and y-z planes for temporal fusion. The procedure of 2D fusion is shown in Fig. 3(b). First, the point feature inputs (H'_{t-1}, F_t^{in})

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on February 18,2025 at 13:15:05 UTC from IEEE Xplore. Restrictions apply.

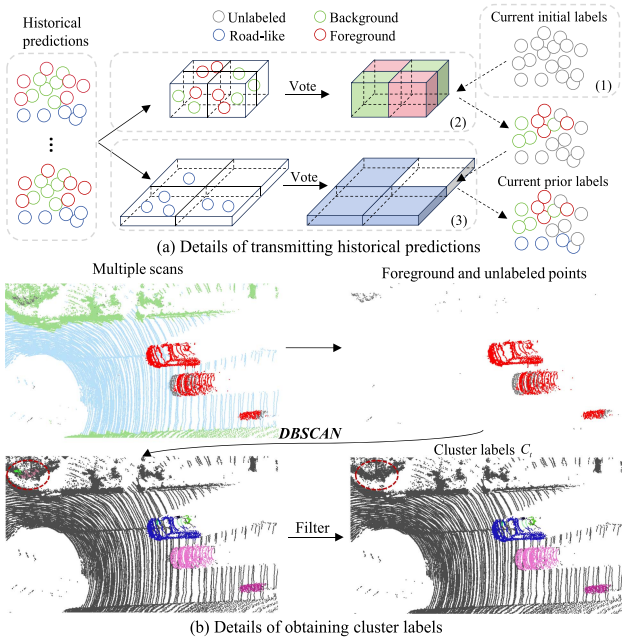

Fig. 4. The illustration of cluster label generation. We first leverage voxels to transfer historical semantic predictions to the current points, and then the DBSCAN is used to generate the clusters of foreground objects.

are projected into 2D grids along a specific coordinate axis. Later, we average point features within the same grid and get 2D features $(\hat{H}'_{t-1}, \hat{F}_t)$ of size $H \times W \times D$. Next, they are combined along channel dimension, and a 1×1 convolution is used to perform feature fusion. The 2D features are then back-projected to the corresponding 3D points to replace the original features. Finally, by carrying out the mentioned step across different views, we could embed historical knowledge and obtain enhanced features $H_t \in \mathbb{R}^{N_p \times D}$, thereby reducing information loss due to occlusion.

C. Cluster-Based Branch

Most semantic segmentation networks [10], [13] typically lack instance-level perception, which will lead to inconsistent semantic predictions for points belonging to the same object (Fig. 1(a) and (b)). To address this, we aim to utilize cluster results from multi-scan as additional information to enhance spatio-temporal consistency in semantic segmentation.

Cluster Label Generation: Due to the continuity of point cloud sequences, we can adopt ego-motion to align past scans with the current points and assign historical predictions to the present frame. Then, for points categorized as foreground, we can employ DBSCAN to segment them into multiple clusters and acquire cluster labels (Fig. 1(e) and (f)).

Specifically, as shown in Fig. 4(a), we transfer historical semantic predictions to the current points by the following steps: (1) *Label Initialization:* Due to focusing on the consistency of foreground segmentation, we map historical predictions to background, foreground, and road-like. Meanwhile, all points in the t frame are initialized as "unlabeled". (2) *Non-ground Label Assignment:* At first, we transfer historical non-ground points to the coordinate system of t frame by a transformation matrix. Next, we separate the 3D space into multiple voxels of size (w, l, h) and feed historical points into corresponding voxels. Through the max-voting operation, the voxel class is assigned based on the most frequent category among its points. Thereafter, we allocate the voxel classes to the current frame based on the coordinate relationships. (3) *Ground Label Assignment:* If translation occurs between two frames, the ground points in the current frame may not have nearby corresponding points from historical frames, resulting in many ground points remaining unlabeled in small voxels of step (2). Thus, we use larger and flatter voxels (w', l', h') to assign road-like labels to "unlabeled" points.

To obtain foreground cluster results that encompass spatio-temporal information, we cluster dense points M_t which is the stacked point clouds of multiple scans as shown in Fig. 4(b). Yet, since the foreground objects may be moving, some dynamic points are still unlabeled after label assignment. Therefore, we retain the points categorized as the foreground and unlabeled across multiple scans, and then process them using DBSCAN to obtain initial cluster results $\widetilde{C}_t = \{c_i\}_{i=1}^{N_c}$, where N_c means the number of clusters. Except for dynamic objects, new background points observed in the current scan may also be classified as unlabeled, leading to some clusters belonging to the background. To tackle this, we loop through the classes of all points within each cluster and retain clusters that contain foreground points. Then, the filtered clusters are denoted as $C_t = \{c_i \mid \exists p_i^j \in c_i \text{ and } L(p_i^j) = \text{"foreground"}\}$, where p_i^j is the j-th point in c_i, and L represents predicted category of each point.

Instance Feature Aggregation: This part aims to gather the features H_t of point-based branch based on cluster labels C_t to acquire instance information. A simple but effective way is to average all point features within the same cluster to yield the cluster features $U_t = \{u_i \in \mathbb{R}^D\}_{i=1}^{N_c}$. Meanwhile, the 3D coordinates of points are also averaged to produce the cluster centers $G_t = \{g_i \in \mathbb{R}^3\}_{i=1}^{N_c}$. However, due to the sparsity or occlusion of point clouds, DBSCAN may separate points of the same object into multiple clusters, resulting in the cluster feature u_i not reflecting the instance information well. Thus, we propose a Temporal Cluster Enhancement (TCE) module to supplement cluster features with neighbors across multi-frame and improve the integrity of cluster information.

In TCE, we project historical cluster centers G_{t-1} to the current coordinate system by transformation matrix T_{t-1}^t and combine it with the current cluster to acquire the new cluster centers $G_{t-1}^t = \{g'_j\}_{j=1}^M$ and corresponding features $U_{t-1}^t = \{u'_j\}_{j=1}^M$, where M represents the total number of clusters across multiple frames. Then, we leverage KNN to search for neighbors $\mathcal{N}(g_i) = \{(g'_j, u'_j) \mid g'_j \in Neighborhood(g_i)\}$ in G_{t-1}^t, which are adjacent to the current cluster centers G_t. For the given cluster $c_i = (g_i, u_i)$, we utilize a linear layer to map feature u_i into *query* q_i. And the feature u'_j of the neighboring clusters $c'_j = (g'_j, u'_j) \in \mathcal{N}(g_i)$ is projected into *key* k_j and *value* v_j vectors. After that, we divide the channel of $v_j \in \mathbb{R}^D$ into h groups ($1 \le h \le D$) and employ Grouped Vector Attention [16] to aggregate cluster features u'_j near c_i, which is denoted as:

$$w_{ij} = \omega(k_j - q_i + \delta_{bias}(g'_j - g_i)), \tag{1}$$

$$u_i^{attn} = \sum_{c'_j}^{\mathcal{N}(g_i)} \sum_{l=1}^{h} \sum_{m=1}^{D/h} \text{Softmax}(W_i)_{jl} v_j^{lD/h+m}, \tag{2}$$

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on February 18,2025 at 13:15:05 UTC from IEEE Xplore. Restrictions apply.

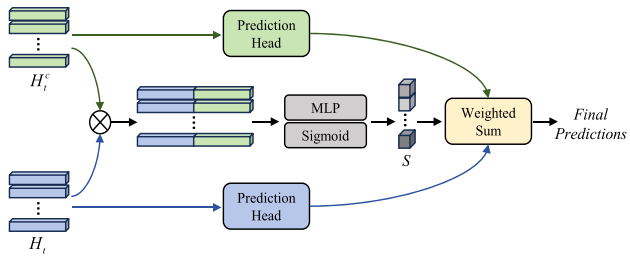

Fig. 5. Illustration of Adaptive Prediction Fusion (APF) module. We adopt different heads to estimate logits for point features from different branches while combining these two features to compute confidence scores. Then, we perform a weighted sum of logits to generate the final prediction results.

where δ_{bias} is the positional encoding function, and W_i is the collection of all w_{ij} for different neighbors c'_j of cluster c_i. $\omega : \mathbb{R}^D \rightarrow \mathbb{R}^h$ means the learnable grouped weight encoding. Meanwhile, enhanced cluster features are formulated as $U'_t = \{u_i^{attn}\}_{i=1}^{N_c}$. In the end, we distribute the cluster features $U'_t \in \mathbb{R}^{N_c \times D}$ to the corresponding foreground points. For residual points, we fill zero as their features and get a final point-wise cluster feature $H_t^c \in \mathbb{R}^{N_p \times D}$ that has the same size as H_t.

D. Point-Cluster Fusion

To combine the semantic features and instance information of two branches and get spatio-temporal consistent segmentation results, we propose an Adaptive Prediction Fusion (APF) module to adaptively merge the predictions of two branches in the point-cluster fusion stage. As illustrated in Fig. 5, for the features (H_t, H_t^c) from different branches, we adopt specific heads to estimate semantic categories and motion states for each point, obtaining the semantic logits (P_{sem}, P_{sem}^c) and motion logits (P_{mov}, P_{mov}^c). Later, to weight the predicted logits from two branches, we join the point features (H_t, H_t^c) along the channel dimension and compute confidence scores $S \in \{S_{sem}, S_{mov}\}$, with values ranging from 0 to 1, through two MLPs that do not share weights.

$$S = \text{Sigmoid}(\text{MLP}(\text{Concat}(H_t, H_t^c))). \tag{3}$$

Afterward, the confidence scores (S_{sem}, S_{mov}) are employed to merge predicted logits of two branches adaptively, which can be represented by the following formula:

$$P_{sem}^{final} = (1 - S_{sem}) \cdot P_{sem} + S_{sem} \cdot P_{sem}^c, \tag{4}$$

$$P_{mov}^{final} = (1 - S_{mov}) \cdot P_{mov} + S_{mov} \cdot P_{mov}^c. \tag{5}$$

E. Loss Function

During the training process, given the ground truth labels, we adopt the predicted semantic logits P_{sem}^{final} and the motion logits P_{mov}^{final} of each point to calculate the losses as follows:

$$\mathcal{L} = \mathcal{L}_{ce}^{sem} + \mathcal{L}_{ls}^{sem} + \mathcal{L}_{ce}^{mov} + \mathcal{L}_{ls}^{mov}, \tag{6}$$

where $\mathcal{L}_{ce}^{sem}$ and $\mathcal{L}_{ce}^{mov}$ are cross-entropy losses for semantic and motion prediction, respectively. $\mathcal{L}_{ls}^{sem}$ and $\mathcal{L}_{ls}^{mov}$ are the Lovasz Softmax Loss [27] for semantic and motion results. This loss function serves as a differentiable surrogate, aiming to optimize the Intersection over Union (IoU) that is used to measure segmentation quality, thereby compensating for the shortcomings of cross-entropy loss in the optimization objective.

IV. Experiment

A. Dataset

SemanticKITTI [31] is a widely used dataset for semantic understanding in outdoor scenes. It utilizes 64-beam LiDAR to collect point clouds and consists of 22 LiDAR sequences, with sequences 00 to 10 as the training set (sequence 08 as the validation set) and sequences 11 to 21 as the testing set. The semantic segmentation task is separated into single-scan (19 categories) only distinguishing object classes and multi-scan (25 categories) extra required to identify motion states of the foreground objects. Besides, the SemanticKITTI-MOS is another benchmark that only determines the dynamic and static states of points. Moreover, nuScenes [32] is composed of 1,000 driving scenes collected by a 32-beam LiDAR sensor and provides 16 semantic classes. Then, following methods [29], [30], we use ground-truth 3D bounding boxes to create 8 moving categories additionally.

B. Evaluation Metric

We adopt the intersection over the union (IoU) to evaluate different methods. The IoU is defined as $\frac{TP}{TP+FP+FN}$, where the TP, FP, and FN denote the true positive, false positive, and false negative of predictions. For the multi-scan benchmark, we adopt mIoU as the evaluation metric, which denotes the IoU of all classes. For the MOS benchmark, we apply the IoU of the moving objects as the evaluation metric.

C. Implementation Details

During the training and testing process, we use three consecutive frames of point clouds as input for SemanticKITTI dataset. For nuScenes dataset, in which the LiDAR operates at 20Hz, we choose three frames with a temporal stride of 2 to better capture object motion. We adopt WaffleIron [14] with $L = 48$ layers as a backbone network. Similar to [14], we downsample point clouds by keeping only one point per voxel of size 10 cm. For the hyperparameters of WaffleIron, we utilize $D = 256$ and a grid resolution ρ of 40 cm for SemanticKITTI, and $D = 384$ with 60 cm grid for nuScenes. For cluster label generation, the voxel size of non-ground label assignment is set to $(0.2\text{ m}, 0.2\text{ m}, 0.2\text{ m})$, while the voxel size of ground label assignment is $(10.0\text{ m}, 10.0\text{ m}, 0.2\text{ m})$. Moreover, we train the network without historical features for 45 epochs using two NVIDIA RTX 4090 GPUs. Afterward, the backbone is frozen, and the residual modules are trained for an additional 45 epochs. The AdamW [33] is adopted to optimize the network with a weight decay of 0.003 and batch size of 6. Besides, our data augmentation strategy includes random flipping, rotation, scaling and instance cutmix with polarmix [14].

D. Evaluation Results

Quantitative Results: As displayed in Table. I and Table. II, we compare our algorithm with other methods on the multi-scan semantic segmentation of SemanticKITTI and nuScenes. The results demonstrate that the proposed 4D-CS achieves state-of-the-art performance in terms of mIoU. Compared to the

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on February 18,2025 at 13:15:05 UTC from IEEE Xplore. Restrictions apply.

TABLE I
THE RESULTS ON THE MULTI-SCAN SEMANTIC SEGMENTATION OF THE SEMANTICKITTI TEST SET

Methods	mIoU(%)	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic sign	car (m)	bicyclist (m)	person (m)	motorcyc. (m)	other-veh. (m)	truck (m)
TemporalLidarSeg [11]	47.0	92.1	47.7	40.9	39.2	35.0	14.4	0.0	0.0	91.8	59.6	75.8	23.2	89.8	63.8	82.3	62.5	64.7	52.6	60.4	68.2	42.8	40.4	12.9	12.4	2.1
TemporalLatticeNet [28]	47.1	91.6	35.4	36.1	26.9	23.0	9.4	0.0	0.0	91.5	59.3	75.3	27.5	89.6	65.3	84.6	66.7	70.4	57.2	60.4	59.7	41.7	51.0	48.8	5.9	0.0
Meta-RangeSeg [12]	49.7	90.8	50.0	49.5	29.5	34.8	16.6	0.0	0.0	90.8	62.9	74.8	26.5	89.8	62.1	82.8	65.7	66.5	56.2	64.5	69.0	60.4	57.9	22.0	16.6	2.6
KPConv [3]	51.2	93.7	44.9	47.2	42.5	38.6	21.6	0.0	0.0	86.5	58.4	70.5	26.7	90.8	64.5	84.6	70.3	66.0	57.0	53.9	69.4	67.4	67.5	47.2	4.7	5.8
Cylinder3D [6]	52.5	94.6	67.6	63.8	41.3	38.8	12.5	1.7	0.2	90.7	65.0	74.5	32.3	92.6	66.0	85.8	72.0	68.9	63.1	61.4	74.9	68.3	65.7	11.9	0.1	0.0
MarS3D [29]	52.7	95.1	49.2	49.5	39.7	36.6	16.2	1.2	0.0	89.9	66.8	74.3	26.4	92.1	68.2	86.0	72.1	70.5	62.8	64.8	78.4	67.3	58.0	36.3	10.0	5.1
Cluster3DSeg [26]	54.7	95.3	55.9	52.9	42.7	38.7	15.5	0.0	3.0	91.4	66.1	76.9	27.8	91.4	66.1	86.5	72.7	71.6	64.0	68.0	81.7	68.2	61.8	46.0	11.2	42.7
MemorySEG [13]	58.3	94.0	68.3	**68.8**	51.3	40.9	27.0	0.3	2.8	89.9	64.3	74.8	29.2	92.2	69.3	84.8	75.1	70.1	65.5	68.5	71.7	74.4	71.7	73.9	15.1	13.6
SVQNet [10]	60.5	96.1	64.4	60.3	40.4	**60.9**	27.4	0.0	0.0	**93.2**	**71.6**	**80.5**	37.0	**93.7**	72.6	**87.3**	**76.7**	72.3	**68.4**	71.0	80.5	72.4	**84.7**	**91.0**	7.5	3.9
2DPASS [20]	62.4	96.2	63.6	63.7	48.2	52.7	**35.4**	**7.9**	**62.0**	89.7	67.4	74.7	**40.0**	93.6	**72.9**	86.2	73.9	71.0	65.0	70.5	82.1	71.2	80.3	73.1	3.8	16.1
WaffleIron [14]	58.4	96.0	**69.0**	66.8	39.9	42.3	33.4	0.4	0.0	90.6	67.9	75.3	27.7	93.3	70.6	86.6	73.6	71.9	63.9	69.0	84.2	**76.5**	70.8	45.5	20.8	24.7
4D-CS (Ours)	**63.7**	**96.7**	66.0	66.5	**62.4**	59.3	33.7	6.7	15.0	90.4	68.3	75.3	32.6	93.4	71.3	87.0	73.9	**72.5**	65.3	**71.3**	**86.0**	72.3	76.6	64.6	**35.5**	**50.9**
Improvements Δ	+5.3	+0.7	-3.0	-0.3	+22.5	+17.0	+0.3	+6.3	+15.0	-0.2	+0.4	+0.0	+4.9	+0.1	+0.7	+0.4	+0.3	+0.6	+1.4	+2.3	+1.8	-4.2	+5.8	+19.1	+14.7	+26.2

(m) indicates moving. The highest IoU for each category is bolded. Δ means comparison with baseline.

TABLE II
THE RESULTS OF MULTI-SCAN SEMANTIC SEGMENTATION ON NUSCENES VALIDATION SET

Methods	mIoU(%)	barrier	bicycle	bus	car	construction	motorcycle	pedestrian	traffic cone	trailer	truck	driveable	other flat	sidewalk	terrain	manmade	vegetation	car (m)	bus (m)	truck (m)	const. (m)	trailer (m)	motor. (m)	bicyc. (m)	person (m)
MarS3D [29]	54.3	70.5	24.7	60.0	**79.9**	32.0	34.9	**51.3**	53.0	10.4	66.0	95.4	59.9	72.7	**75.8**	87.2	**86.1**	66.5	48.0	52.4	0.0	23.1	69.0	9.7	72.7
SegNet4D [30]	57.9	77.4	32.6	63.8	73.8	41.1	44.0	51.2	63.2	42.3	74.2	96.2	69.3	74.2	73.5	64.6	55.9	68.6	51.3	59.4	0.0	27.2	74.3	40.8	72.4
WaffleIron [14]	65.7	78.5	48.3	69.8	72.8	50.6	59.1	49.2	69.9	54.3	56.0	**96.9**	73.6	**75.5**	74.0	87.9	85.4	70.9	62.2	49.4	**0.3**	59.5	89.7	**70.2**	**73.5**
4D-CS (Ours)	**67.3**	**78.8**	**51.7**	**77.5**	78.9	**51.6**	**61.1**	40.4	**70.4**	**56.5**	**75.4**	96.8	**73.7**	75.0	73.7	**88.0**	85.4	**75.2**	**65.7**	**64.0**	0.2	**61.3**	**90.4**	55.2	69.1

(m) indicates moving.

TABLE III
PERFORMANCE COMPARISON ON THE VALIDATION AND TEST SET OF SEMANTICKITTI-MOS

Methods	IoU_M (Validation 08)	IoU_M (Test 11-21)
MotionSeg3D [34]	71.4	64.9
4DMOS [22]	77.2	65.2
LMNet [35]	67.1	62.5
RVMOS* [23]	71.2	73.3
InsMOS* [36]	73.2	70.6
InsMOS*† [36]	69.4	75.6
MotionBEV [37]	76.5	69.7
MF-MOS* [24]	76.1	76.7
4D-CS*(Ours)	**80.9**	**83.5**

† indicates training on both SemanticKITTI and KITTI-road datasets, while * denotes methods using semantic labels.

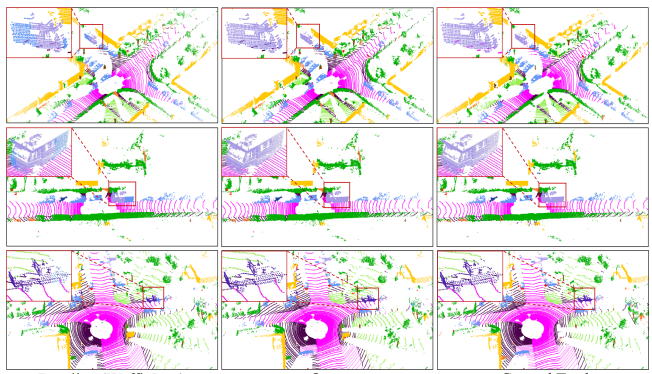

Fig. 6. The visualization of semantic segmentation results on the validation set of SemanticKITTI. We highlight the area that reflects the advantages of our method, which is displayed in the upper left corner.

baseline [14], we achieve a significant improvement for large foreground objects with a 22.5% increase for trucks, 17.0% for other vehicles, 26.2% for dynamic trucks, and 14.7% for moving other vehicles. In Table. II, our method also achieves IoU improvement of most foreground objects, especially for large ones, such as trucks and buses. This demonstrates that explicit clustering priors can help the network to focus on the complete spatial information of objects, rather than relying on the local features obtained from a limited receptive field as other algorithms do, resulting in better foreground object segmentation results. Moreover, we compare the performance of our approach on the MOS benchmark of the SemanticKITTI in Table. III. Our approach surpasses the state-of-the-art work MF-MOS [24] by 6.8% IoU_M on the testing set. This validates that transmitting historical features at both point and instance levels can not only improve the integrity of segmentation but also enhance the model's ability to identify object motion states.

Qualitative Comparisons: Semantic qualitative results are illustrated in Fig. 6. It shows that the segmentation results of the baseline network for large objects are prone to truncation due to lacking the ability for instance perception. In contrast, our method can achieve consistent segmentation results after introducing cluster information. Additionally, for the moving qualitative results displayed in Fig. 7, the baseline model still struggles to segment moving objects completely, whereas our method successfully achieves this. Overall, our approach has a stronger capability to accurately and consistently recognize the categories and motion states of foreground objects.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on February 18,2025 at 13:15:05 UTC from IEEE Xplore. Restrictions apply.

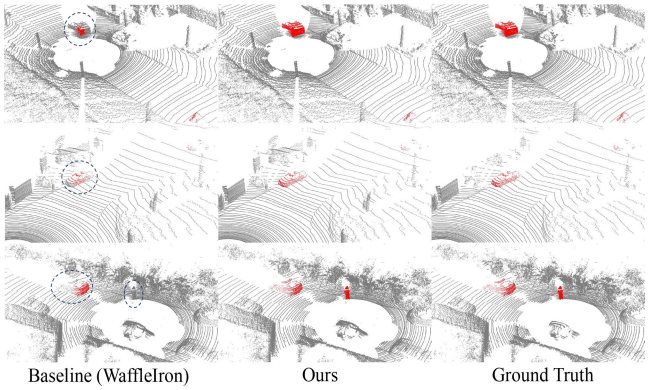

Fig. 7. The visualization of moving object segmentation on the validation set of SemanticKITTI. We mark the poor predictions of baseline with blue dashed circles.

TABLE IV
THE EFFECT OF DIFFERENT MODULES ON SEMANTICKITTI VALIDATION SET

Baseline	APF	TCE*	MTF	TCE	mIoU (%)	IoU_M (%)
✓					56.4	76.7
✓	✓				57.1	79.2
✓	✓	✓			57.3	79.6
✓	✓	✓	✓		57.7	80.0
✓	✓		✓	✓	**58.0**	**80.9**

mIoU is the mean IoU of all classes on the semantic segmentation. IoU_M is the IoU of moving objects.

E. Ablation Studies

In this section, we conduct comprehensive ablation experiments on the validation set of the SemanticKITTI dataset.

Model Components: As shown in Table. IV, we first employ backbone [14] to extract features and directly use the output features to generate predictions, resulting in a baseline multi-scan semantic mIoU of 56.4% and 76.7% IoU of the moving objects. After that, we introduce cluster labels to gather point features and adopt Adaptive Prediction Fusion (APF) module to optimize the prediction results, leading to the improvement of 0.7% mIoU and 2.5% IoU_M. It proves the proposed APF can effectively utilize cluster knowledge to improve instance-level perception and enhance the accuracy of semantic predictions. Since only foreground objects have motion states, this instance-level perception of the foreground significantly improves the IoU of moving objects. Later, we exploit TCE* (i.e., Temporal Cluster Enhancement (TCE) module without historical cluster features), leading to better performance and proving the effectiveness of merging nearby cluster features. In the end, we supply historical features into our pipeline. On the one hand, we adopt Multi-View Temporal Fusion (MTF) module to combine past point features, increasing both mIoU and the IoU_M by 0.4%. On the other hand, we adopt the TCE module with historical cluster features and improve mIoU by 0.3% and IoU_M by 0.9%. These results indicate that fusing historical priors can effectively enhance the current point and cluster features, resulting in better segmentation results.

Dual-branch Fusion: As shown in Table. V, we compare different strategies for merging information from two branches. It is worth noting that our network contains a TCE* module and a dual-branch fusion module with different strategies in this part. First, we directly overwrite point features with cluster features, leading to a performance drop compared to using a weighted sum. This is because the foreground point segmentation depends on cluster features, where an error will lead to all points of a cluster being predicted incorrectly. Then, we concatenate the features from different branches and use MLP for fusion, but the results are inferior to the weighted sum. Besides, compared to the adaptive weighted fusion of predictions in APF, we attempt a hard fusion of prediction logits from different branches, but it also leads to an unideal result. This proves that our proposed weight-based soft fusion can avoid damage to the original results caused by some poor cluster features and achieve better predictions.

TABLE V
MORE DETAILED ABLATION EXPERIMENTS ON THE MODULES OF 4D-CS ON SEMANTICKITTI VALIDATION SET

Module	Strategy	mIoU (%)	IoU_M (%)
APF	Direct Overwrite	55.6	77.4
	Feature Fusion	56.8	79.3
	Unweighted Sum	57.1	78.7
	Weighted Sum	**57.3**	**79.6**
MTF	w/o MTF	57.3	79.6
	Only BEV View	57.6	79.7
	Multiple Views	**57.7**	**80.0**

Multi-view Temporal Feature Fusion: In this section, our network is comprised of the TCE* and APF modules, while we compare historical feature fusion on the single-view and multi-view. In Table. V, compared to the situation without temporal fusion, the method that incorporates historical features shows improvements in both multi-scan semantic mIoU and IoU_M of moving objects. In particular, our proposed multi-view fusion strategy achieves the best performance improvement, proving that multi-view fusion could more effectively reduce information loss and convey temporal cues more completely.

F. Runtime and Memory

In this section, we employ an NVIDIA RTX 4090 GPU to measure the inference time for multi-scan semantic segmentation on the SemanticKITTI dataset. With three point cloud frames, our baseline method (WaffleIron) takes 117 ms and occupies 8.2 GB of memory. In comparison, our proposed algorithm requires 151 ms for network processing and 5 ms for cluster label generation, utilizing 9.9 GB of memory.

V. CONCLUSION

In this letter, we analyze the limitations of existing multi-scan segmentation methods and propose a novel dual-branch structure, which aims to use cluster information to improve spatio-temporal consistency of segmentation results. We first fuse temporal point features by the multi-view representation. Then, we utilize cluster labels to integrate point features and acquire instance information, which is refined by combining neighboring clusters across multiple frames. Finally, we fuse information from two branches adaptively to optimize the class prediction of each point, thereby boosting the consistency of segmentation. The experiments show that our 4D-CS exceeds the previous state-of-the-art multi-scan semantic and moving object segmentation methods.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on February 18,2025 at 13:15:05 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "SuMa++ : Efficient LiDAR-based semantic SLAM," in *Proc. 2019 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2019, pp. 4530–4537.

[2] S. Garg et al., "RoboHop: Segment-based topological map representation for open-world visual navigation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2024, pp. 4090–4097.

[3] 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 *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 6411–6420.

[4] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet : Deep hierarchical feature learning on point sets in a metric space," *Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5105–5114.

[5] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.

[6] X. Zhu et al., "Cylindrical and asymmetrical 3D convolution networks for lidar segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 9934–9943.

[7] X. Lai, Y. Chen, F. Lu, J. Liu, and J. Jia, "Spherical transformer for LiDAR-based 3D recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pp. 17545–17555, 2023.

[8] H.-X. Cheng, X.-F. Han, and G.-Q. Xiao, "Cenet: Toward concise and efficient LiDAR semantic segmentation for autonomous driving," in *Proc. 2022 IEEE Int. Conf. Multimedia Expo.*, 2022, pp. 01–06.

[9] L. Kong et al., "Rethinking range view representation for LiDAR segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 228–240.

[10] X. Chen, S. Xu, X. Zou, T. Cao, D.-Y. Yeung, and L. Fang, "SVQNet: Sparse voxel-adjacent query network for 4D spatio-temporal LiDAR semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 8569–8578.

[11] F. Duerr, M. Pfaller, H. Weigel, and J. Beyerer, "LiDAR-based recurrent 3D semantic segmentation with temporal memory alignment," in *Proc. 2020 Int. Conf. 3D Vis.*, 2020, pp. 781–790.

[12] S. Wang, J. Zhu, and R. Zhang, "Meta-RangeSeg: LiDAR sequence semantic segmentation using multiple feature aggregation," *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 9739–9746, Oct. 2022.

[13] E. Li, S. Casas, and R. Urtasun, "MemorySeg: Online LiDAR semantic segmentation with a latent memory," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 745–754.

[14] G. Puy, A. Boulch, and R. Marlet, "Using a waffle iron for automotive point cloud semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 3379–3389.

[15] M. Ester et al., "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discov. Data Mini*, 1996, vol. 96, pp. 226–231.

[16] X. Wu, Y. Lao, L. Jiang, X. Liu, and H. Zhao, "Point transformer V2: Grouped vector attention and partition-based pooling," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 33330–33342, 2022.

[17] Y. Zhang et al., "PolarNet: An improved grid representation for online LiDAR point clouds semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9601–9610.

[18] H. Tang et al., "Searching efficient 3D architectures with sparse point-voxel convolution," in *Proc. Eur. Conf. Computer Vis.*, Springer, 2020, pp. 685–702.

[19] 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 *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 16024–16033.

[20] X. Yan et al., "2DPASS: 2D priors assisted semantic segmentation on LiDAR point clouds," in *Proc. Eur. Conf. Comput. Vis.*, Springer, 2022, pp. 677–695.

[21] X. Wu et al., "TASeg: Temporal aggregation network for LiDAR semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 15311–15320.

[22] B. Mersch, X. Chen, I. Vizzo, L. Nunes, J. Behley, and C. Stachniss, "Receding moving object segmentation in 3D LiDAR data using sparse 4D convolutions," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 7503–7510, Jul. 2022.

[23] J. Kim, J. Woo, and S. Im, "RVMOS: Range-view moving object segmentation leveraged by semantic and motion features," *IEEE Robot. Automat. Lett.*, vol. 7, no. 3, pp. 8044–8051, Jul. 2022.

[24] J. Cheng et al., "Mf-mos: A motion-focused model for moving object segmentation," in *proc. IEEE Int. Conf. Robot. Automat.*, 2024, pp. 12499–12505.

[25] D. Lu, J. Zhou, K. Y. Gao, J. Du, L. Xu, and J. Li, "Dynamic clustering transformer network for point cloud segmentation," *Int. J. Appl. Earth Observation Geoinformation*, vol. 128, 2024, Art. no. 103791.

[26] T. Feng, W. Wang, X. Wang, Y. Yang, and Q. Zheng, "Clustering based point cloud representation learning for 3D analysis," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 8283–8294.

[27] M. Berman, A. R. Triki, and M. B. Blaschko, "The Lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks," in *Proc. IEEE Conf. Comput. Vis. pattern Recognit.*, 2018, pp. 4413–4421.

[28] P. Schutt, R. A. Rosu, and S. Behnke, "Abstract flow for temporal semantic segmentation on the permutohedral lattice," in *Proc. 2022 Int. Conf. Robot. Automat.*, 2022, pp. 5139–5145.

[29] J. Liu, C. Chang, J. Liu, X. Wu, L. Ma, and X. Qi, "MarS3D: A plug-and-play motion-aware model for semantic segmentation on multi-scan 3D point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 9372–9381.

[30] N. Wang et al., "SegNet4D: Effective and efficient 4D LiDAR semantic segmentation in autonomous driving environments," 2024, *arXiv:2406.16279*.

[31] J. Behley et al., "SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 9297–9307.

[32] H. Caesar et al., "nuscenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11621–11631.

[33] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Representations*, 2019.

[34] J. Sun et al., "Efficient spatial-temporal information fusion for LiDAR-based 3D moving object segmentation," in *Proc. 2022 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 11456–11463.

[35] X. Chen et al., "Moving object segmentation in 3D LiDAR data: A learning-based approach exploiting sequential data," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 6529–6536, Oct. 2021.

[36] N. Wang, C. Shi, R. Guo, H. Lu, Z. Zheng, and X. Chen, "Insmos: Instance-aware moving object segmentation in LiDAR data," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2023, pp. 7598–7605.

[37] B. Zhou, J. Xie, Y. Pan, J. Wu, and C. Lu, "MotionBEV: Attention-aware online LiDAR moving object segmentation with bird's eye view based appearance and motion features," *IEEE Robot. Automat. Lett.*, vol. 8, no. 12, pp. 8074–8081, 2023.

Authorized licensed use limited to: Shenyang Institute of Automation. Downloaded on February 18,2025 at 13:15:05 UTC from IEEE Xplore. Restrictions apply.