

Knowledge Distillation for 3D LiDAR Segmentation

1st Huancheng Xiao

School of Artificial Intelligence
Shenzhen Technology University
Shenzhen, China
2310413007@email.szu.edu.cn

2nd Qiang Li

School of Artificial Intelligence
Shenzhen Technology University
Shenzhen, China
liqiang1@sztu.edu.cn

3rd Ye Gu

School of Artificial Intelligence
Shenzhen Technology University
Shenzhen, China
guye@sztu.edu.cn

Abstract—Accurate semantic segmentation of LiDAR point clouds is critical for autonomous driving, yet single-modal approaches often struggle with complex environments due to limited depth and geometric information. While multi-modal fusion of LiDAR and camera data has shown promise, existing fusion based models suffer from high computational costs and inadequate fine-grained details. This paper proposes a novel framework that enhances 3D point cloud segmentation by integrating 2D image features and 1D serialized point cloud features through cross-attention. Our approach leverages space-filling curves to preserve local spatial structures in serialized point clouds and employs dynamic grid pooling to adaptively capture multi-scale features. By distilling 2D and 1D knowledge into the 3D backbone, our method achieves superior performance, significantly reducing computational overhead. Extensive experiments on the SemanticKITTI dataset demonstrate that our approach outperforms state-of-the-art methods, achieving a mean IoU of 0.733, with notable improvements in segmenting tiny size objects like bicycles and motorcycles.

Index Terms—point cloud segmentation, knowledge distillation, space-filling curves

I. INTRODUCTION

3D point cloud semantic segmentation has been one of the key technologies in the field of autonomous driving. With the development of deep learning, the segmentation accuracy relying solely on LiDAR has been greatly improved. LiDAR can only capture textureless and sparse data. Therefore, Lidar-image fusion methods have gradually gained favor among researchers. Since images can provide rich color and texture information. However, existing fusion based models still have limitations: 1) Currently, most fusion methods are implemented through multi-task or cascaded architectures, which require simultaneous processing of image and point cloud data during inference, consuming more computing resources. 2) In the fusion stage, the voxelization and 3D sparse convolution feature extraction practices can cause the loss of local fine-grained information. In contrast, we propose a LiDAR point cloud segmentation method that fuses 3D point clouds, 2D images, and 1D serialized features only through cross-attention. The 3D backbone is trained using 2D and 1D serialized features through knowledge distillation.

A. Point-based Segmentation

PointNet [1] pioneers direct point-wise segmentation via MLP-based networks. While follow-ups (e.g., PointNet++ [2]) work for indoor data, most fail to scale to large outdoor

scenes due to computational/memory limits—even state-of-the-art KPConv [3] (with spatial kernel-based convolution) relies on radial balanced sampling (risking information loss). Point-based methods also suffer from inefficient local neighbor search.

B. Voxel-based Segmentation

This branch converts unstructured point clouds to regular voxels for 3D convolution-based learning (e.g., early VoxNet [4]), but struggles with accuracy-efficiency balance. Recent optimizations include lightweight 3D convolution (reducing costs for real-time use) and non-uniform voxel grids—Cylinder3D [5] stands out, using asymmetric residual modules to retain target geometry while cutting computation. AF2S3Net [6] further boosts performance via dual attention and anisotropic loss for fine-structure restoration.

C. Multi-Modal Segmentation

Due to the numerous limitations of single-modal models, the academic community has increasingly focused on the direction of multi-modal fusion. El *et al.* [7] addressed the under-explored 3D semantic segmentation for LiDAR data in autonomous driving, proposing an RGB-LiDAR fusion framework that converts RGB images into LiDAR-compatible polar-grid mapping. Zhuang *et al.* [8] fuse point clouds with perspective views, but most only use one-to-one fusion (underutilizing RGB) and fail with inaccurate calibration. In contrast, the proposed method enables adaptive fusion and mitigates point-pixel misalignment via a learnable cross-modal module. Liu *et al.* [9] proposed a unified multi-modal LiDAR segmentation network that fuses RGB images with three LiDAR point cloud views (point, voxel, range view).

II. METHODS

A. Framework Overview

The proposed framework is shown in Fig. 1. The entire workflow is mainly divided into feature extraction, feature fusion and multi-scale distillation learning. The feature extraction process of 2D images first passes through the embedding layer for preprocessing, and then the multi-scale image features are successively obtained by the 2D encoder and decoder. The feature extraction procedure of 3D point clouds is similar to that of 2D images, and this process is accompanied by the voxelization process of the point clouds. For the feature

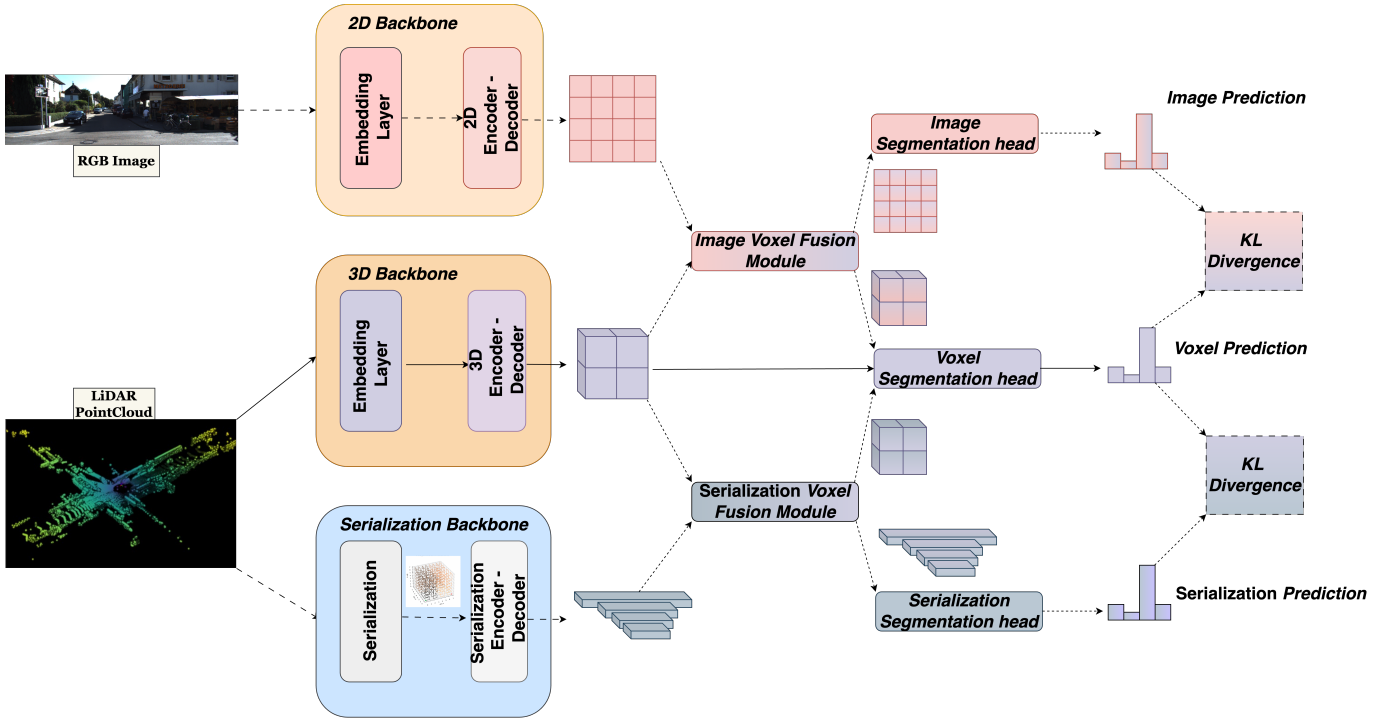


Fig. 1: **Framework Overview.** The main framework includes three segmentation branches. Among them, the solid line represents the workflow of training and inference, while the dashed line only exists in the training phase.

extraction of serialized point clouds, before the serialization enters the codec for feature extraction, the 3D point cloud is first transformed into a 1D spatial sequence through a spatial curve. Through these three individual feature extractor, we obtained three types of feature information. Further, these multi-scale feature information is fused and knowledge is distilled. During the distillation process, it not only makes full use of the prior knowledge of texture and color information of 2D images, but also uses serialized features to compensate for the loss of spatial local fine-grained information during the voxelization process of 3D point clouds. In contrast to the method that merely utilizes modal fusion, 2D branches and serialized branches are only involved in the training process, thereby reducing the computational overhead significantly.

B. Point Cloud Serialization

To preserve the fine-grained information of the spatial structure of the point cloud, we employ point cloud serialization. The core idea of point cloud serialization is to convert unstructured point clouds into structured sequences. The space-filling curve [10] algorithm can map high-dimensional spaces to one-dimensional indicators. Essentially, a space-filling curve is a path that can traverse all grid points in a high-dimensional discrete space and maintain the proximity relationship between points in the space to a certain extent. Fig. 2 shows a 3D schematic diagram of the Hilbert [11] space filling curve. Another relatively common one is the Z-order space filling curve. Mathematically, it can be precisely defined by a bijective function, denoted as $\varphi : \mathbb{Z} \rightarrow \mathbb{Z}^n$.

Here, $\mathbb{Z}$ represents a set of one-dimensional integer indices, corresponding to the linear traversal order of the curve, n denotes the dimension of the target space. In the context of 3D point cloud processing, n is typically set to 3 to match the 3D coordinates of point clouds. Meanwhile, this function can be flexibly extended to higher-dimensional spaces. This bijective property implies that each grid point in the high-dimensional space can correspond to a unique one-dimensional integer index, and vice versa, providing a rigorous mathematical foundation for the conversion of high-dimensional data to low-dimensional indices. Due to the bijective nature of the space-filling curve, there exists an inverse mapping $\varphi^{-1} : \mathbb{Z}^n \rightarrow \mathbb{Z}$, through this mapping, we can project the point $p_i \in \mathbb{R}^3$ onto a discrete space with a grid size of $g \in \mathbb{R}$. The operation can be defined as $\varphi^{-1} \left(\left\lfloor \frac{p}{g} \right\rfloor \right)$. By sorting the obtained codes, we obtain the 1D sequence of point clouds arranged in accordance with spatial curves.

C. Model Architecture

Feature Extraction. As shown in Fig. 1, in our model, we mainly use three independent networks to extract different types of features. For 2D images, we used the ResNet34 to construct 2D feature extraction backbone. For 3D feature extraction, we use Cylinder3D [5] as 3D backbone. The extraction of 1D sequence features is achieved through the aforementioned point cloud serialization. We utilize the bijective property of spatial curves to obtain the encoding numbers of points in the point cloud within the curve, and sort the numbers in ascending order to obtain the 1D point cloud

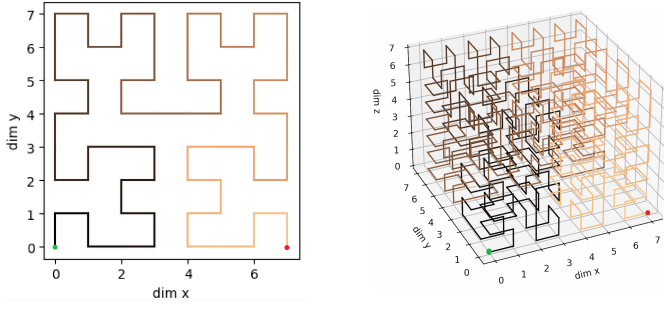


Fig. 2: **Hilbert Curve**. (a) The 3-stage order 2D Hilbert curve. (b) The 3-stage 3D Hilbert curve. The green dots represent the starting point of the traversal, and the red dots represent the ending point of the traversal.

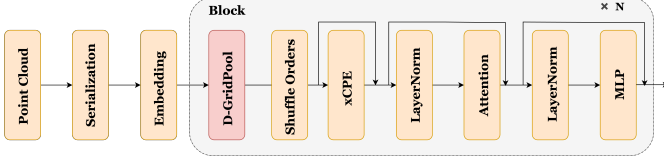


Fig. 3: **Serialization Feature Extractor**. Unlike PTv3, we introduce a dynamic grid pooling layer (D-GridPool) to perform multi-scale feature extraction on the point cloud sequence.

sequence. To enable the model to better capture sequence features, we introduced group attention, performing non-overlapping grouping on the original point cloud sequence and implementing an attention mechanism among these groups. Since the group size is fixed, but not all points in the point cloud can be divided evenly by the group size, in response to this situation, we use points from adjacent groups to supplement the groups with insufficient size. Predecessors have fully explored the effectiveness of this serialized point cloud grouping [12]. In PTv3 [11], Grid Pool aggregates local point cloud features via a globally uniform mesh partitioning strategy. However, point clouds inherently exhibit spatial heterogeneity, with substantial variations in point density and geometric complexity across regions—dense areas (e.g., object surfaces) contain rich details, while sparse areas (e.g., open backgrounds) have simple structures. This uniform grid fails to balance detail preservation and computational efficiency: oversized grids cause fine structures (e.g., edges, textures) loss in dense regions, whereas undersized grids introduce redundant computations in sparse regions.

To address this limitation, we propose dynamic grid pooling (D-GridPooling) as an improvement. As shown in Fig. 3, D-Gridpooling has replaced the original GridPooling. D-GridPooling first computes metrics (e.g., average k-nearest neighbor distance of each point) to characterize local density. It then dynamically determines mesh size based on this density: small meshes are allocated to dense, complex areas (e.g., object edges, surfaces) for detail retention, while large meshes are assigned to sparse, simple areas (e.g., open planes, sparse backgrounds) to reduce redundancy. Finally, local features are aggregated via attention weighting within meshes of varying sizes.

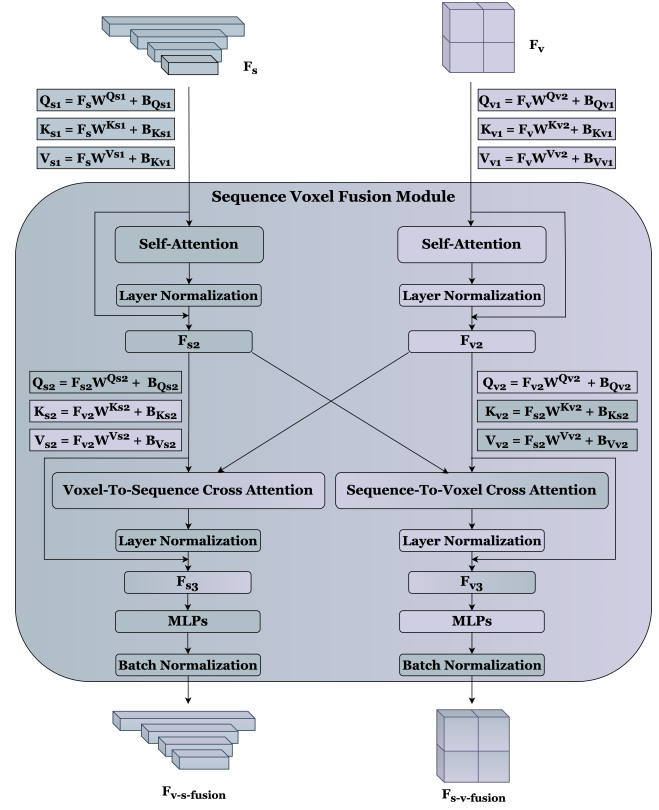


Fig. 4: **Serialization Voxel Fusion Module**.

Modal Fusion. After going through three feature extraction backbones, we obtained three different types of feature data, where F_v represents voxel features, F_r represents image features, and F_s represents sequence features. We will fully fuse the image features and sequence features with the voxel features respectively through two modules, namely Sequence-Voxel Fusion Module and Image-Voxel Fusion Module. Because the main tasks of the two Fusion modules are similar, we mainly introduce the Sequence-Voxel Fusion Module here. Its working principle is presented in Fig 4. Firstly, we will perform linear projection on the sequence feature F_s and the voxel feature F_v to generate the Query, Key, and Value of the self-attention module, which are $Q_{s1}, Q_{v1}, K_{s1}, K_{v2}, V_{s1}$ and V_{v1} respectively. Then, these will be processed through the attention mechanism and layer normalization to obtain the enhanced sequence feature F_{s2} and voxel feature F_{v2} . The next stage is the fusion of features. Here, it is divided into independent cross-attention from sequence to voxel and cross-attention from voxel to sequence. We conducted attention learning from sequence to voxel using F_{v2} as the Key and Value, and F_{s2} as the Query. After layer normalization and MLP processing, the sequence feature F_{s3} with enhanced F_{v2} features was obtained. The cross-attention from voxel to sequence, in turn, yields the voxel feature F_{v3} enhanced by the F_{s2} feature. After feature fusion, we finally obtained four features enhanced by fusion. They are respectively marked as

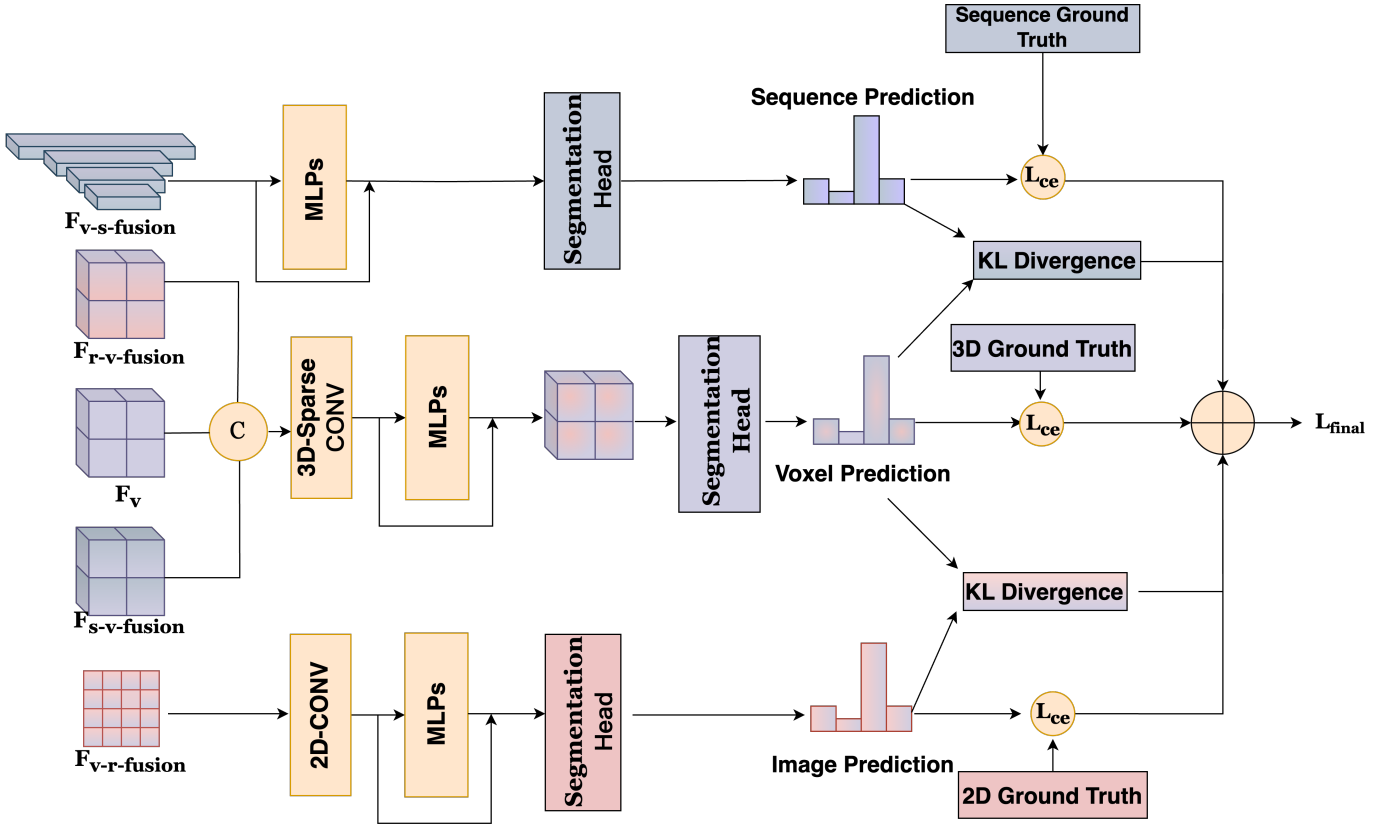


Fig. 5: **Knowledge Distillation.** The knowledge distilling process from 2D image features and 1D point cloud sequence features to 3D models. Among them, L_{ce} represents the prediction results of the three modes and the Cross-Entropy Loss corresponding to the ground truth. These loss values will eventually be combined with the KL divergence value as the final loss L_{final} for each calculation.

$F_{v-s-fusion}, F_{v-r-fusion}, F_{r-v-fusion}$, and $F_{s-v-fusion}$.

$$Q_\theta = F_s W_{Q_\theta} + B_{Q_\theta} \quad (1)$$

$$K_\theta = F_s W_{K_\theta} + B_{K_\theta} \quad (2)$$

$$V_\theta = F_s W_{V_\theta} + B_{V_\theta} \quad (3)$$

$$\text{Attn}(Q, K, V) = \text{Softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad (4)$$

$$F_{s2} = \text{LayerNorm}(\text{Attn}(Q_{s1}, K_{s1}, V_{s1})) + F_s \quad (5)$$

$$F_{v2} = \text{LayerNorm}(\text{Attn}(Q_{v1}, K_{v1}, V_{v1})) + F_v \quad (6)$$

$$F_{s3} = \text{LayerNorm}(\text{Attn}(Q_{s2}, K_{s2}, V_{s2})) + F_{s2} \quad (7)$$

$$F_{v3} = \text{LayerNorm}(\text{Attn}(Q_{v2}, K_{v2}, V_{v2})) + F_{v2} \quad (8)$$

$$F_{v-s-fusion} = \text{BatchNorm}(\text{MLP}(F_{s3})) \quad (9)$$

$$F_{s-v-fusion} = \text{BatchNorm}(\text{MLP}(F_{v3})) \quad (10)$$

We will utilize these features to segment the predictions and to transfer the 2D and 1D branches to 3D branches. The entire workflow and principle are illustrated in Fig 5. We divide fusion features into three independent segmentation heads to generate prediction results for these four features. For image features and sequence features, we designed multi-layer MLP pooling and softmax to achieve result prediction of P_r and P_s . As for the voxel features, we first concatenate $F_{r-v-fusion}$ and $F_{v-f-fusion}$ on the channel, and then obtain

the prediction result P_v through multiple layers of MLP and softmax.

$$P_r = \text{Softmax}(\text{MLP}(F_{v-r-fusion})) \quad (11)$$

$$P_s = \text{Softmax}(\text{MLP}(F_{v-s-fusion})) \quad (12)$$

$$P_v = \text{Softmax}(\text{MLP}(F_{r-v-fusion} \oplus F_{s-v-fusion} \oplus F_v)) \quad (13)$$

Inspired by 2DPass, we do not simply fuse the prediction results to obtain the final result, as this does not achieve the purpose of discarding 2D and serializing branches during reasoning. Our approach is to generate 2D ground truth and 1D sequence ground truth respectively through 3D ground truth. For 1D sequence ground truth, when performing point cloud serialization, we recorded the mapping of each point, so 1D sequence ground truth can be obtained easily. For 2D ground truth, we rely on the camera calibration parameters and calibration values contained in the dataset to perform projection transformation on 3D ground truth to obtain it.

$$G_r = \Pi_{\text{proj}}(G_v, \mathbf{K}, \mathbf{T}) \quad (14)$$

$$G_s = \Gamma_{\text{seq}}(G_v, \mathcal{M}) \quad (15)$$

In order for the 2D branch and serialized branch to have an impact on the final model during training, we perform weighted summation of the ground truth of the three modalities and the loss generated by the prediction. Meanwhile, we also

TABLE I: The Comparison of our method and others on **Semantickitti**

Methods	mIoU	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign
PointNet [1]	0.146	0.463	0.013	0.003	0.001	0.008	0.002	0.002	0.0	0.616	0.158	0.357	0.014	0.414	0.129	0.310	0.046	0.176	0.024	0.037
MPF [13]	0.555	0.934	0.302	0.383	0.261	0.285	0.481	0.461	0.181	0.906	0.623	0.745	0.306	0.885	0.597	0.835	0.597	0.692	0.497	0.581
FidNet [14]	0.595	0.939	0.547	0.489	0.276	0.239	0.623	0.598	0.237	0.906	0.591	0.758	0.267	0.889	0.605	0.845	0.644	0.690	0.533	0.628
Cylinder3D [5]	0.618	0.961	0.542	0.476	0.386	0.450	0.651	0.635	0.136	0.912	0.622	0.752	0.187	0.896	0.616	0.854	0.697	0.693	0.626	0.647
RangeViT [15]	0.640	0.954	0.558	0.435	0.298	0.421	0.639	0.582	0.381	0.931	0.702	0.800	0.325	0.920	0.690	0.853	0.706	0.712	0.608	0.647
CENet [16]	0.647	0.919	0.586	0.503	0.406	0.423	0.689	0.659	0.435	0.903	0.609	0.751	0.315	0.910	0.662	0.845	0.697	0.700	0.615	0.676
JS3C-Net [17]	0.660	0.958	0.593	0.529	0.543	0.460	0.695	0.654	0.399	0.888	0.619	0.721	0.319	0.925	0.708	0.845	0.698	0.680	0.607	0.687
TransUPR [18]	0.682	0.933	0.536	0.681	0.586	0.637	0.700	0.706	0.523	0.904	0.702	0.748	0.421	0.891	0.628	0.628	0.646	0.694	0.564	0.630
TPV-IGKD [19]	0.693	0.967	0.633	0.658	0.546	0.631	0.765	0.807	0.416	0.900	0.603	0.754	0.291	0.911	0.663	0.859	0.742	0.703	0.638	0.691
AF2S3Net [6]	0.697	0.945	0.654	0.868	0.392	0.411	0.807	0.804	0.743	0.913	0.688	0.725	0.535	0.879	0.632	0.702	0.685	0.537	0.615	0.710
RPVNet [20]	0.703	0.976	0.684	0.687	0.442	0.611	0.759	0.744	0.434	0.934	0.703	0.807	0.333	0.935	0.721	0.865	0.751	0.717	0.648	0.614
SDSeg3D [21]	0.704	0.974	0.587	0.542	0.549	0.652	0.702	0.744	0.522	0.909	0.694	0.767	0.419	0.932	0.711	0.861	0.743	0.711	0.654	0.706
GASN [22]	0.707	0.969	0.658	0.580	0.593	0.610	0.804	0.827	0.463	0.898	0.662	0.746	0.301	0.923	0.696	0.873	0.730	0.725	0.661	0.716
PVKD [23]	0.712	0.970	0.679	0.693	0.535	0.602	0.751	0.735	0.505	0.918	0.709	0.775	0.410	0.924	0.694	0.865	0.738	0.719	0.649	0.658
CMDFusion [24]	0.716	0.968	0.630	0.674	0.531	0.612	0.766	0.815	0.772	0.901	0.669	0.753	0.307	0.915	0.680	0.855	0.744	0.699	0.645	0.675
PMformer [25]	0.716	0.965	0.645	0.647	0.565	0.555	0.782	0.821	0.649	0.895	0.669	0.747	0.394	0.928	0.703	0.863	0.722	0.706	0.637	0.715
MDC-Seg [26]	0.724	0.971	0.695	0.718	0.586	0.609	0.774	0.766	0.538	0.915	0.709	0.770	0.429	0.926	0.703	0.867	0.742	0.717	0.651	0.664
2DPass [27]	0.729	0.970	0.636	0.634	0.611	0.615	0.779	0.813	0.741	0.897	0.674	0.747	0.400	0.935	0.729	0.862	0.739	0.710	0.650	0.704
RangeFormer [28]	0.733	0.967	0.694	0.737	0.599	0.662	0.781	0.759	0.581	0.924	0.730	0.788	0.424	0.923	0.701	0.866	0.733	0.728	0.664	0.666
Ours	0.733	0.965	0.689	0.690	0.620	0.629	0.832	0.838	0.460	0.910	0.708	0.771	0.462	0.931	0.723	0.865	0.745	0.723	0.673	0.709

take the KL divergence values of P_v and P_r , as well as P_v and P_s respectively as the distillation losses, thereby enhancing the one-way knowledge distillation of 2D information and 1D serialized information to the 3D network.

$$\mathcal{L}_{ce}(P, G) = - \sum_c G(c) \log P(c) \quad (16)$$

$$\mathcal{L}_{sup} = \lambda_v \mathcal{L}_{ce}(P_v, G_v) + \lambda_r \mathcal{L}_{ce}(P_r, G_r) + \lambda_s \mathcal{L}_{ce}(P_s, G_s) \quad (17)$$

$$\text{KL}(P \parallel Q) = \sum_c P(c) \log \frac{P(c)}{Q(c)} \quad (18)$$

$$\mathcal{L}_{distill} = \text{KL}(P_v \parallel P_r) + \text{KL}(P_v \parallel P_s) \quad (19)$$

$$\mathcal{L}_{total} = \mathcal{L}_{sup} + \alpha \mathcal{L}_{distill} \quad (20)$$

III. EXPERIMENTS

DataSet and Metric. The dataset we used in the experiment is the semantic segmentation dataset from the KITTI Vision Odometry Benchmark — Semantickitti [29]. There are 23,201 point clouds for training and 20,351 for testing in the dataset. As with most of the work, we use the average intersection and union ratio of all categories specified in [29] to evaluate our method which is calculated as:

$$\text{IoU}_i = \frac{\text{TP}_i}{\text{TP}_i + \text{FP}_i + \text{FN}_i}, \quad (21)$$

$$\text{mIoU} = \frac{1}{N} \sum_{i=1}^N \text{IoU}_i, \quad (22)$$

where TP_i , FP_i , and FN_i represent true positive, false positive, and false negative predictions for class i , and the mIoU is the mean value of IoU_i .

Results on Semantickitti. As shown in Tab. I, we compared our method with others' on the Semantickitti

Benchmark. Our method has been improved in the overall mIoU. Compared with 2DPass that uses both 2D and 3D features, our mIoU has increased by 0.04. The model outperform 2DPass in majority of the classes.

TABLE II: Comparison with and without sequence branches on **Semantickitti**

Methods	mIoU	Single Class Result		
		bicycle	bicyclist	motorcycle
Without Sequence	0.721	0.677	0.765	0.684
With Sequence	0.733	0.689	0.838	0.690

Influence of Sequence Branch. To verify the role of serialized branches in our method, we also conduct some ablation experiments. As shown in Tab. II, discarding serialized branches reduces mIoU by 0.012. The results for small types with complex details have also decreased accordingly. This proves that the serialized branch effectively distills fine-grained knowledge of the local space to the 3D model.

TABLE III: Comparison of GridPool and D-GridPool on **Semantickitti**

Methods	mIoU	Single Class Result		
		bicycle	bicyclist	motorcycle
GridPool	0.730	0.665	0.758	0.681
D-GridPool	0.733	0.689	0.838	0.690

Influence of Dynamic Grid Pooling. As we mentioned earlier, in our serialized feature extraction work, we introduced the D-GridPool branch to replace the GridPool of the original backbone. We also conducted some ablation experiments to verify the effectiveness of D-GridPool. Under the condition that other factors remain unchanged, we compared the performance of D-GridPool and the original GridPool. As can be seen from Tab. III, D-GridPool has increased mIoU by 0.003. Although the improvement of mIoU is not dramatic, it still indicates that D-GridPool remains a feasible attempt.

IV. CONCLUSION

This paper proposes a model that utilizes two-dimensional image information and 1D serialized point clouds to enhance the performance of 3D models in LiDAR point cloud semantic segmentation. By leveraging the cross-attention mechanism, space-filling curves, and dynamic grid pooling, rich texture and color information can be extracted from the two-dimensional images, and fine spatial information can be extracted from the 1D point cloud sequence, effectively improving the semantic segmentation performance of 3D networks. The model has been verified on the SemanticKITTI dataset: its mIoU reaches 0.733, which is 0.004 higher than 2DPass. The ablation experiments in the paper further confirm that the 1D serialized branch (increasing by 0.12) and D-GridPool (increasing by 0.03) play a key role in enhancing fine segmentation and balancing efficiency.

ACKNOWLEDGMENT

This project was supported by Shenzhen Science and Technology Program No. JCYJ20220818102215034, Research Promotion Project of Key Construction Discipline in Guangdong Province No. 2022ZDJS112.

REFERENCES

- [1] 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*, pp. 652–660, 2017.
- [2] 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.
- [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 *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6411–6420, 2019.
- [4] 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)*, pp. 922–928, IEEE, 2015.
- [5] X. Zhu, H. Zhou, T. Wang, F. Hong, Y. Ma, W. Li, H. Li, and D. Lin, "Cylindrical and asymmetrical 3d convolution networks for lidar segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9939–9948, 2021.
- [6] R. Cheng, R. Razani, E. Taghavi, E. Li, and B. Liu, "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*, pp. 12547–12556, 2021.
- [7] K. El Madawi, H. Rashed, A. El Sallab, O. Nasr, H. Kamel, and S. Yogamani, "Rgb and lidar fusion based 3d semantic segmentation for autonomous driving," in *2019 IEEE intelligent transportation systems conference (ITSC)*, pp. 7–12, IEEE, 2019.
- [8] Z. Zhuang, R. Li, K. Jia, Q. Wang, Y. Li, and M. Tan, "Perception-aware multi-sensor fusion for 3d lidar semantic segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 16280–16290, 2021.
- [9] Y. Liu, R. Chen, X. Li, L. Kong, Y. Yang, Z. Xia, Y. Bai, X. Zhu, Y. Ma, Y. Li, et al., "Uniseg: A unified multi-modal lidar segmentation network and the openpcseg codebase," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 21662–21673, 2023.
- [10] G. Peano, "Sur une courbe, qui remplit toute une aire plane," in *Arbeiten zur Analysis und zur mathematischen Logik*, pp. 71–75, Springer, 1990.
- [11] X. Wu, L. Jiang, P.-S. Wang, Z. Liu, X. Liu, Y. Qiao, W. Ouyang, T. He, and H. Zhao, "Point transformer v3: Simpler faster stronger," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4840–4851, 2024.
- [12] Z. Liu, X. Yang, H. Tang, S. Yang, and S. Han, "Flatformer: Flattened window attention for efficient point cloud transformer," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1200–1211, 2023.
- [13] Y. A. Alnaggar, M. Afifi, K. Amer, and M. ElHelw, "Multi projection fusion for real-time semantic segmentation of 3d lidar point clouds," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pp. 1800–1809, 2021.
- [14] Y. Zhao, L. Bai, and X. Huang, "Fidnet: Lidar point cloud semantic segmentation with fully interpolation decoding," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4453–4458, IEEE, 2021.
- [15] A. Ando, S. Gidaris, A. Bursuc, G. Puy, A. Boulch, and R. Marlet, "Rangevit: Towards vision transformers for 3d semantic segmentation in autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5240–5250, 2023.
- [16] H.-X. Cheng, X.-F. Han, and G.-Q. Xiao, "Cenet: Toward concise and efficient lidar semantic segmentation for autonomous driving," in *2022 IEEE international conference on multimedia and expo (ICME)*, pp. 01–06, IEEE, 2022.
- [17] X. Yan, J. Gao, J. Li, R. Zhang, Z. Li, R. Huang, and S. Cui, "Sparse single sweep lidar point cloud segmentation via learning contextual shape priors from scene completion," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, pp. 3101–3109, 2021.
- [18] Z. Yu, M. Chen, Z. Zhang, S. You, R. Rao, S. Agarwal, and F. Ren, "Transupr: A transformer-based plug-and-play uncertain point refiner for lidar point cloud semantic segmentation," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5864–5869, IEEE, 2023.
- [19] J.-C. Li, J.-G. Lu, M. Wei, H.-Y. Kang, and Q.-H. Zhang, "Tpv-igkd: Image-guided knowledge distillation for 3d semantic segmentation with tri-plane-view," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 8, pp. 10405–10416, 2024.
- [20] 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*, pp. 16024–16033, 2021.
- [21] J. Li, H. Dai, and Y. Ding, "Self-distillation for robust lidar semantic segmentation in autonomous driving," in *European conference on computer vision*, pp. 659–676, Springer, 2022.
- [22] M. Ye, R. Wan, S. Xu, T. Cao, and Q. Chen, "Efficient point cloud segmentation with geometry-aware sparse networks," in *European conference on computer vision*, pp. 196–212, Springer, 2022.
- [23] Y. Hou, X. Zhu, Y. Ma, C. C. Loy, and Y. Li, "Point-to-voxel knowledge distillation for lidar semantic segmentation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8479–8488, 2022.
- [24] J. Cen, S. Zhang, Y. Pei, K. Li, H. Zheng, M. Luo, Y. Zhang, and Q. Chen, "Cmdfusion: Bidirectional fusion network with cross-modality knowledge distillation for lidar semantic segmentation," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 771–778, 2023.
- [25] X. Li, X. Tan, Z. Zhang, Y. Xie, and L. Ma, "Point mask transformer for outdoor point cloud semantic segmentation," *Computational Visual Media*, pp. 497–511, 2025.
- [26] X. Ouyang, X. Qian, Y. Zhang, Y. Shen, G. Wang, and W. Liu, "Mdcseg: Multi-directional convolution-based semantic segmentation for lidar point clouds," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–7, IEEE, 2025.
- [27] X. Yan, J. Gao, C. Zheng, C. Zheng, R. Zhang, S. Cui, and Z. Li, "2dpass: 2d priors assisted semantic segmentation on lidar point clouds," in *European conference on computer vision*, pp. 677–695, Springer, 2022.
- [28] L. Kong, Y. Liu, R. Chen, Y. Ma, X. Zhu, Y. Li, Y. Hou, Y. Qiao, and Z. Liu, "Rethinking range view representation for lidar segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 228–240, 2023.
- [29] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "Semantickitti: A dataset for semantic scene understanding of lidar sequences," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9297–9307, 2019.