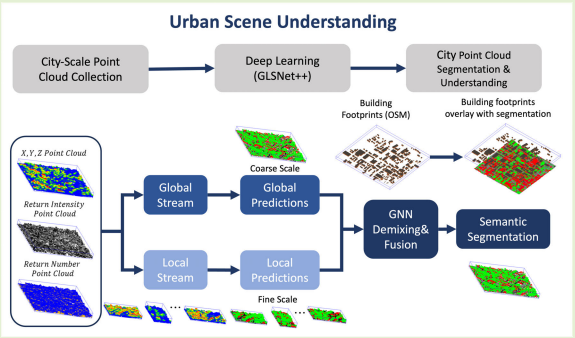
GLSNet++: Global and Local-Stream Feature Fusion for LiDAR Point Cloud Semantic Segmentation Using GNN Demixing Block

Rina Bao, Kannappan Palaniappan, *Senior Member, IEEE*, Yunxin Zhao, *Life Senior Member, IEEE*, and Guna Seetharaman, *Fellow, IEEE*

***Abstract*—Semantic point cloud segmentation is a critical task in 3-D computer vision, offering valuable contextual information for navigation, cartography, landmarks, object recognition, and building modeling. We developed global and local stream deep network (GLSNet++), an innovative deep learning architecture for robust context-dependent 3-D point cloud segmentation. GLSNet++ uniquely combines dual streams of global and local feature manifolds to capture multiscale contextual and structural information, addressing challenges due to highly varying object sizes in urban scenes. To effectively and efficiently refine mixed class labels from cross-scale global and local streams, GLSNet++ incorporates a novel graph neural network (GNN)-based demixing block (GDB) for accurately resolving class membership near voxel boundaries with spatial context-dependent feature fusion. We validate GLSNet++ on the IEEE DFT4 LiDAR dataset, achieving competitive city-scale semantic segmentation that can be extended to more classes, higher-resolution point clouds, and larger geographic regions. GLSNet++ exhibits strong generalization when tested on an independent LiDAR dataset from Columbia, Missouri evaluated using OpenStreetMap (OSM).**



***Index Terms*—3-D semantic segmentation, building information modeling (BIM), ensemble stacking, feature fusion, geographical information system (GIS), hyperspectral unmixing, point clouds.**

I. Introduction

THE rapid progress of diverse 3-D geospatial data acquisition techniques has established point clouds as the primary source for obtaining extensive urban 3-D geospatial information across scale [1]. These point clouds are indispensable for smart city wide area modeling [2], [3], [4], autonomous driving [5], urban planning [6], [7], visual odometry for robotics [8], [9], agriculture [10], and other applications [11], [12], [13], [14], [15]. To effectively use 3-D point clouds in dynamic perception for practical applications, precise and efficient semantic 3-D object segmentation of large-scale point clouds is essential. However, volumetric city-scale point cloud segmentation is a challenging problem due to the high degree of variation in object scale, mixed classes, complex object shapes, high density of similar objects, and high computational cost for volumetric processing. Geospatial point clouds have millions to billions of 3-D points. Object sizes may be very large, requiring a segmentation approach that captures long-range global structural and contextual information. At the same time, some classes include very small compositional objects, such as the architectural details of buildings, walls and fences, leaves of vegetation (shrubs, bushes, or trees), power lines and poles, railway tracks which require the segmentation to capture fine-scale local information to delineate salient object boundaries.

Existing approaches when applied to large-scale 3-D point cloud semantic segmentation exhibit several limitations.

1) They need to split the point clouds into multiple blocks [16], [17], [18], [19], [20], [21], [22], [23], [24],

Manuscript received 23 July 2023; revised 26 October 2023; accepted 7 November 2023. Date of publication 8 February 2024; date of current version 2 April 2024. This work was supported in part by the U.S. Army Research Laboratory under Grant W911NF-1820285 and in part by the Engineering Research and Development Center—Information Technology Laboratory (ERDC-ITL) under Contract W912HZ23C0041. The associate editor coordinating the review of this article and approving it for publication was Dr. Geethu Joseph. *(Corresponding authors: Rina Bao; Kannappan Palaniappan.)*

Rina Bao was with the Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO 65211 USA. She is now with Boston Children's Hospital and Harvard Medical School, Boston, MA 02115 USA (e-mail: rina.bao@childrens.harvard.edu).

Kannappan Palaniappan and Yunxin Zhao are with the Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO 65211 USA (e-mail: pal@missouri.edu).

Guna Seetharaman is with the Naval Research Laboratory, Washington, DC 20375 USA (e-mail: guna@ieee.org).

Digital Object Identifier 10.1109/JSEN.2023.3345747

[25] due to the computational cost and limited network capacity, which leads to the loss of long-range contextual and structural information.

2) They use random sampling [26] to capture long-range information, which results in the loss of certain geometric information especially in more intricate scenes, increased noise, and occlusion between different objects, making it insufficient to capture local information within the point cloud. However, relationships between local structures become crucial for scene reasoning, as it links semantic inference to effectively discriminate between many classes that exhibit significant intraclass variance.
3) Although volumetric methods [27], [28], [29] can be utilized by employing regular convolutions if they are efficiently designed and trained, and have demonstrated efficacy in coarse-grained feature learning applications, these networks are limited to providing voxel-level predictions, and their voxelization sampling procedure combines multiple raw points into a single voxel, resulting in ambiguous or erroneous predictions near object boundaries where voxels contain a group of 3-D points from a mixture of different classes.

The above limitations point to the need for investigating a novel approach for efficiently fusing both *global and local, structural and contextual* feature information for accurate large-scale LiDAR point cloud segmentation. We believe that designing a network with modules specialized for global and local information processing can help address such a challenging problem, where the global stream captures long-range and large-scale structural and contextual information, while the local stream captures small-scale structural and contextual information. The two stream ensemble should be aggregated in a complementary way. In our previous paper, we proposed a two-stream network architecture, global and local streams deep network [12] (GLSNet), with max pooling aggregation for large point cloud segmentation. The global branch processes a full set of voxels of each LiDAR point cloud for accurate classification over large areas, while the local branch processes a (tile) subset of points each time for accurate classification over small areas, followed by a max pooling fusion operation to integrate the complementary geometric information across scales. Specifically, the global branch is a voxel-based network, inheriting its advantages of efficient coarse feature learning, while the local branch is a graph-based network, inheriting advantages of dedicated fine feature learning. However, using a basic max pooling layer in GLSNet for aggregating the two streams constrained the deep architecture's ability to fully harness the potential of fusing multiscale features, limiting overall performance of the two-stream stacked ensemble.

In this work, we investigate effective methods for synergistically integrating global and local features using spatial contextual information across scale. We capitalize on the global-local information learning approach of GLSNet and extend it significantly in three directions.

1) In network feature stream fusion ensemble architecture, we consider not only parallel network branches of global–local information streams, called GLSNet-Parallel, but also a global-to-local information flow, called GLSNet-Cascade, where a global network module first provides a coarse-scale semantic segmentation, which is followed by a local network module to refine the global branch produced semantic segmentation.
2) We propose GLSNet++, a novel context-dependent graph-based method for information fusion or ensemble aggregation. In information fusion, GLSNet++ takes the global and local prediction results as input and minimizes the fused prediction errors of semantic segmentation 3-D point class labels based on multiple scales and multiple receptive fields using a novel graph neural network (GNN) based demixing block (GDB) that incorporates multiscale context particularly near object 3-D boundaries.
3) We consider real application scenarios where the object labels for point cloud semantic segmentation are hard to acquire, and we propose a benchmarking workflow using building footprints from OpenStreetMap (OSM) as the ground truth for evaluations on such unlabeled LiDAR data. In Fig. 1, we show the point-cloud data for Columbia, Missouri, that was collected by our local

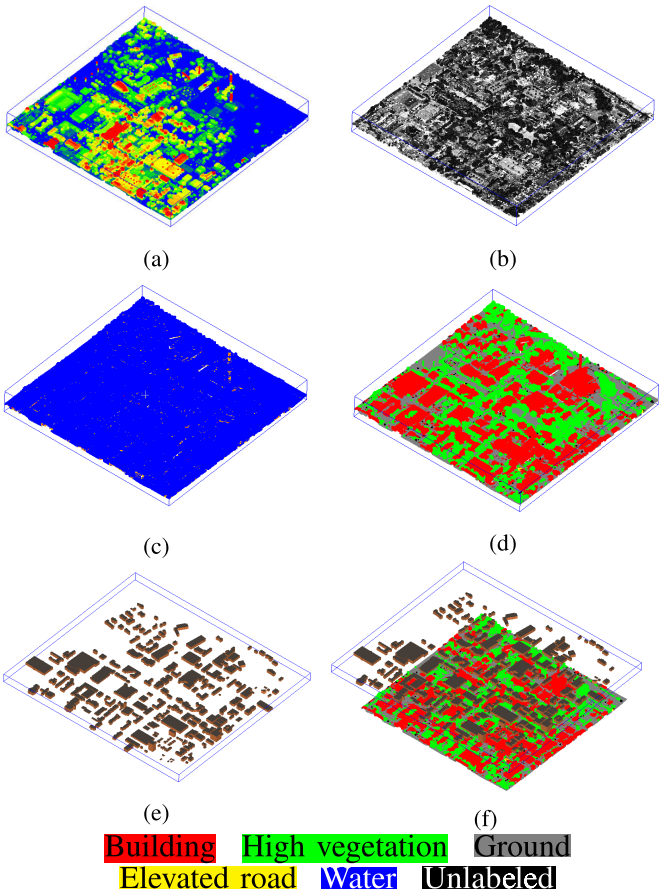


Fig. 1. LiDAR point cloud benchmark for Columbia, Missouri (COU-Orbit30) showing: (a) colored height map (z-value), (b) LiDAR return intensity (0–255), (c) LiDAR return number (0–4) with zero in blue, (d) LiDAR point cloud semantic segmentation using the proposed GLSNet++, (e) building footprints from OSM, and (f) GLSNet++ segmentation result overlaid with OSM building footprints in an orthographic projection.

collaborating team. Even though the point clouds lack accurate labels, the predictions made using point-cloud segmentation exhibit strong alignment with the building footprints extracted from OSM.

Our key contributions are threefold.

1) We introduce a novel network, called GLSNet++, to address the challenges of segmenting objects with large variations urban-scale point cloud semantic segmentation. Dual stream ensemble stacking effectively and efficiently combines global and local information, leveraging a unique GDB with lightweight parameters to fuse global and local features. To the best of our knowledge, this approach has not been explored in previous research on city-scale LiDAR point clouds.
2) We propose a novel benchmarking approach for evaluating the generalization power of segmentation algorithms on new point clouds when ground-truth labels are unavailable. This workflow provides a new and practical method for evaluating the performance of semantic 3-D point cloud segmentation models on real-world data by utilizing building footprints from crowd-sourced OSM without relying solely on (single expert) ground-truth annotations.
3) We did extensive testing, and our experimental results demonstrate that GLSNet++ achieves competitive performance on airborne LiDAR point clouds. The model's strong generalization power is showcased through its ability to effectively segment point clouds from different (unseen) airborne LiDAR datasets, illustrating the advantages, and effectiveness for large-scale urban scenarios.

The subsequent parts of this work are organized as follows. Section II reviews related work in point cloud segmentation. Section III describes different fusion network structures for combining global and local branches, along with the design of our proposed GLSNet++ ensemble stacking with demixing aggregation or fusion block. Section IV describes experimental datasets and implementation details. Section V first presents quantitative and qualitative visualization results of our GLSNet++ method on a public benchmark; then provides a procedure for establishing a benchmark with unlabeled LiDAR point clouds in real application scenarios to evaluate semantic segmentation accuracy across geographical regions. Lastly, we demonstrate the generalization power of GLSNet++ through experimental results along with Conclusions in Section VI.

II. Related Work

We provide an overview of point cloud segmentation techniques, focusing on deep learning methods for large-scale point cloud segmentation. We categorize the popular deep neural network methods for 3-D point cloud segmentation into three groups below.

A. Voxel-Based Networks

Voxel-based methods depend on a regular representation based on voxels, which transforms unordered points into organized 3-D voxel grids. Subsequently, 3-D/2-D convolutions are employed for feature learning [20], [30], [31], [32], [33], [34]. Çiçek et al. [35] proposes a 3-D U-Net structured network for point cloud segmentation. VoxelNet [31] integrates a volumetric occupancy grid representation with a supervised 3-D convolutional neural network (3-D CNN) for point cloud segmentation. PointGrid [20] designs a point quantization network to facilitate learning local geometry shapes. To reduce the computation cost of 3-D convolution, Graham and van der Maaten [29] propose highly efficient convolution named submanifold sparse convolution (SSC), and the subsequently improved SSC network (SSCN) [36] can process a full 3-D point cloud at one time with strong performance. Recent works along this line explored the specialized design of convolution kernels for 3-D points [21], [37], [38]. However, voxel-based approaches are constrained in their ability to offer predictions at the granularity of individual voxels. Besides, regular strategies for capturing long-range information for voxel-based methods, such as using deeper networks and increasing kernel size, will undoubtedly increase computation cost, and adding many pooling layers will cause local information loss.

B. Graph-Based Neural Networks

GNNs or graph convolutional networks (GCNs) have been widely explored in point cloud segmentation [39], [40], [41]. PointNet [16], and its follow-up works fall in this category. PointNet learns from unordered point clouds by point-wise encoding and aggregation through a global max pooling, but its capacity is insufficient in capturing contextual information. Thus, the follow-up work PointNet++ [17] proposes a hierarchical architecture to capture local geometric details at different scales to overcome the weakness of PointNet. Huang et al. [42] introduce a recurrent slice network to model local dependency information. PointSIFT [24] works on an orientation-encoding unit to capture different orientation representations of points. PointHop [23] focuses on an explainable learning method for point cloud classification. PointCNN [19] proposes a $\mathcal{X}$-Conv layer to exploit certain canonical ordering of points. Dynamic graph CNN (DGCNN) [43] suggests semantically grouping points method by dynamically updating relation graphs. Superpoint graphs (SPG) [44] adaptively partition point clouds by contextual relationship encoded SPG. PartNet [45] proposes a top-down recursive decomposition strategy for shape segmentation. PMFRNet [46] proposes a multiscale feature fusion refinement structured neural network with channel attention module focusing on detailed features and domain attention module strengthening the connection of contextual semantic information. Lian et al. [47] proposes DPNet, a dense PointNet++ with multiple supervision and post-processed by applying a grid map-based correction and model fusion. Jia et al. [48] propose to add a point global attention mechanism on top of PointSIFT to improve the segmentation performance on point clouds. SqueezeSeg [49] uses convolutional neural networks to predict point cloud segmentation and refines it by a conditional random field. PolarNet [50] introduces a unique polar bird's-eye-view representation that deviates from conventional spherical or bird's-eye-view projections to evenly distribute points across grid

cells within a polar coordinate system, thereby indirectly aligning a segmentation network's focus with the long-tailed distribution of points along the radial axis. Zhao et al. [22] introduce a new method to extract contextual features from local neighborhoods in a point cloud with a designed adaptive feature adjustment module to find the interaction between points. Point2Sequence [25] focuses on learning 3-D shape features by capturing fine-grained contextual information to explore contextual information in the local regions. RandLA-Net [26] proposes a lightweight network to infer point semantics using random point sampling directly. For graph-based methods, most existing studies need to slice a point cloud into multiple subsets and process one subset at a time when applied to large scale point clouds, which inevitably weakens their global contextual and structural reasoning ability.

C. Global and Local Fusion

A few papers have focused on global and local information aggregation. FusionNet [51] proposes a voxel-based mini-PointNet point cloud representation and a novel feature aggregation module, that combines both voxel aggregation with fine-grain point-wise feature learning. In [52], local and global structures are leveraged for point cloud segmentation through contextual point representations, employing gated fusion, a graph PointNet module, with spatial-wise and channel-wise attention strategies to produce accurate semantic labels for each point. A novel context-aware deep network for large-scale point cloud segmentation is proposed in [53], using a local feature aggregation module, a global context aggregation module, and a context-aware upsampling module. However, these networks are either confined to individually sliced blocks (tiles), leading to the loss of long-range information, or they lack a specific design that imposes global structural constraints across blocks. This results in either an inadequate representation or a loss of long-range structural and contextual information.

Compared with voxel-based networks, GLSNet++ utilizes a voxel-based network as one branch to capture coarse-level information using a large voxel size but with lower computational cost. Compared with graph-based networks, GLSNet++ utilizes it as another branch to capture fine-scale spatial information. Compared with most global–local fusion methods, our global information represents structural and contextual multilevel fusion across sliced point cloud blocks instead of within each block, meaning their global information is still our local information. In summary, we propose GLSNet++ to use global and local feature embedding ensemble streams to process different levels of contextual and structural information in point clouds. We further design a novel demixing block to perform global and local information decomposition and aggregation for 3-D point cloud segmentation.

III. Methodology

In this section, we first describe the 3-D point cloud semantic segmentation problem, then briefly review the global branch and local branch network ensemble architectures, and finally, we elaborate on our three network design schemes for global and local information fusion or aggregation.

A. Point Cloud Semantic Segmentation

Let $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_i, \ldots, \mathbf{s}_N\}$ be the set of feature vectors of N points in a point cloud. For LiDAR point clouds we use

$$\mathbf{s}_i = \{x_i, y_i, z_i, I_i, R_i\} \tag{1}$$

where $\mathbf{s}_i$ is a 5-D vector with $\{x_i, y_i, z_i\}$ being the 3-D *spatial location* of the point in world or relative geometric coordinates, I_i the measured LiDAR *return intensity*, and R_i the LiDAR *return number*. The semantic segmentation task is given a feature set $\mathbf{S}$ of a point set, predict the corresponding class label set $\mathbf{Q}$. The label set $\mathbf{Q}$ is defined as the set of one-hot vectors, with one for each point, and $\mathbf{Q} \in N \times \mathbb{R}^C$, where C is the total number of semantic classes. The estimated $\mathbf{Q}$ is denoted by $\widehat{\mathbf{Q}}$ with

$$\widehat{\mathbf{Q}} = \mathbf{F}(\mathbf{S}) \tag{2}$$

where $\mathbf{F}$ is the learned nonlinear function in a deep network that predicts semantic segmentation labels $\widehat{\mathbf{Q}}$.

B. Global and Local Branches

In Fig. 2, we illustrate our proposed deep learning architecture consisting of a *global branch* and a *local branch* from GLSNet [12]. The *global branch* is designed to capture long-range structural and contextual information with the following considerations: 1) sparse voxelized grids are taken as the input because they preserve point clouds structure, unlike scattered and unordered point set, and a sparse voxelization that employs a large voxel grid radius and aggregation of input point clouds facilitates efficient global information capture without needing a very deep network and 2) a UNet structured SSCN (SSCN-U) is employed in global feature representation learning for semantic segmentation due to the high efficiency of sparse submanifold convolution operation [29]. In short, the global branch first voxelizes an entire point cloud, and then performs SSCN-U on the sparse voxelized point cloud grids. The *local branch* is designed to capture local contextual information and handle fine-grained point cloud segmentation. The sophisticated network PointNet++ serves this purpose well and is taken for this branch. Specifically, the local branch partitions an entire point cloud into overlapped tiles and employs PointNet++ to operate on each tile (subset of points) to segment local areas.

The two branches extract geometric information in a complementary way and are trained separately with the supervision of point cloud segmentation ground truth. We define $\mathbf{P}_{\text{Global}} \in N \times \mathbb{R}^C$ as the output probability of the global branch $\mathbf{F}_{\text{Global}}$, and $\mathbf{P}_{\text{Local}} \in N \times \mathbb{R}^C$ as the output probability of the local branch $\mathbf{F}_{\text{Local}}$.

C. Fusion of Global and Local Streams

We explore three ensemble aggregation methods for global and local information fusion using parallel streams, cascaded streams and parallel streams with label demixing block as shown in Fig. 3. In GLSNet-Parallel [Fig. 3(a)] the global branch and local branch work concurrently in parallel, where training and inference are in parallel, and the outputs integrated through max pooling. In GLSNet-Cascade [Fig. 3(b)] the

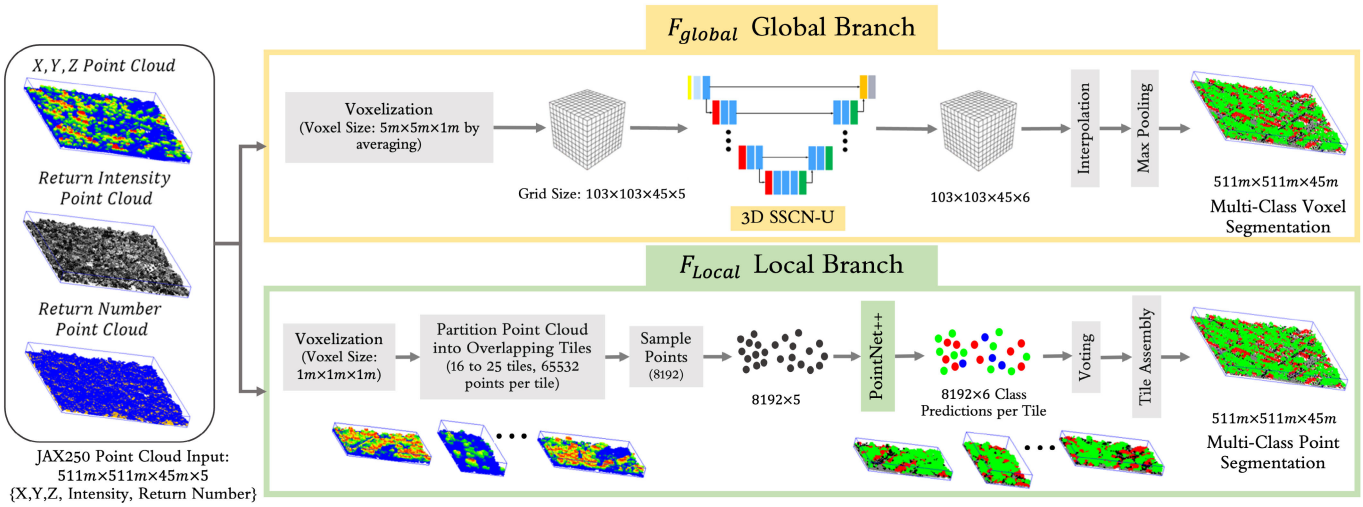


Fig. 2. Dual stream ensemble backbone in GLSNet++ incorporates a global 3-D-voxel branch (3-D SSCN-U, top row) and a parallel local 3-D-point branch (PointNet++, bottom row) that extract coarse-scale and fine-scale geometric information from the input 3-D point cloud. The global and local streams are trained in parallel with independent global and local supervision.

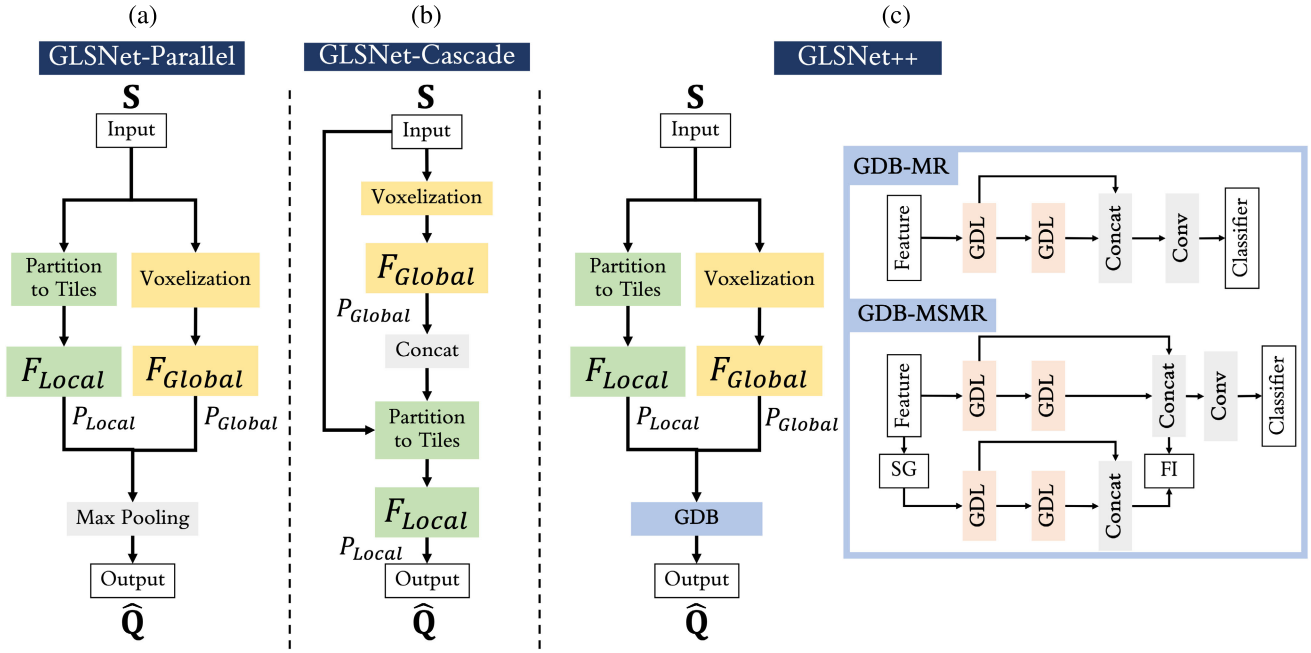


Fig. 3. Explorations of different methods for aggregating global and local branch ensemble stacking: (a) GLSNet-Parallel, (b) GLSNet-Cascade, and (c) GLSNet++. The proposed GLSNet++ structure uses parallel feature embedding (global and local) fusion followed by GNN Demixing Block (GDB) (blue box) of two types: multi-receptive field GDB (top: GDB-MR) and multiscale multi-receptive field GDB (bottom: GDB-MSMR).

global and local network modules work in tandem through a cascade structure, following the coarse to fine refinement framework. Finally, in GLSNet++ [Fig. 3(c)] this architecture uses a novel GDB for multi-scale (MS) and multi-receptive (MR) field context-dependent fusion of global and local information.

1) GLSNet-Parallel: GLSNet-Parallel employs parallel global-local branches and uses max pooling to leverage the complementary information from these two streams to improve the performance, as shown in Fig. 3(a). As a point-set to label set mapping function $\mathbf{F}$, max pooling is performed over the two prediction vectors of the global branch network $\mathbf{P}_{\text{Global}}$ and the local branch network $\mathbf{P}_{\text{Local}}$. The final prediction GLSNet-Parallel is defined by the operation

$$\mathbf{F}_{\text{parallel}}(\mathbf{S}) = \max\left\{\mathbf{F}_{\text{Global}}(\mathbf{S}), \mathbf{F}_{\text{Local}}(\mathbf{S})\right\}. \tag{3}$$

2) GLSNet-Cascade: The GLSNet-Cascade facilitates global-to-local information refinement, where the global module provides a coarse-level semantic segmentation prediction for a given point cloud, and the local module refines the global prediction of class labels. The structure of GLSNet-Cascade is shown in Fig. 3(b), where the input to the local module in GLSNet-Cascade is $[\mathbf{S}, \mathbf{P}_{\text{Global}}]$, with $[\cdot]$ denoting concatenation. The operation of GLSNet-Cascade

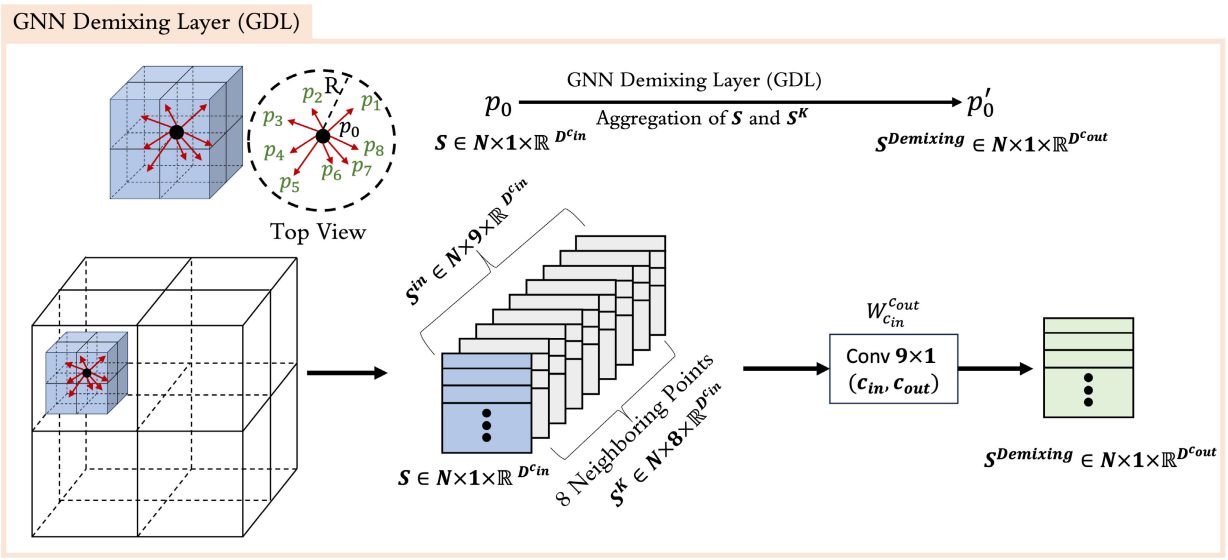


Fig. 4. GDL incorporates local point cloud topology for capturing spatial context in unstructured grids.

is defined as

$$\mathbf{F}_{\text{cascade}}(\mathbf{S}) = \mathbf{F}_{\text{Local}}([\mathbf{S}, \mathbf{F}_{\text{Global}}(\mathbf{S})]). \tag{4}$$

D. GLSNet++ With GDB for Context-Dependent Fusion

The GLSNet++ workflow is illustrated in Fig. 3(c). This design is rooted in our observation that the segmentation errors in GLSNet-Parallel are mainly in regions where global and local branches have different class predictions typically around object boundaries. Therefore, we design a GDB to help demix the class predictions from the two ensemble branches in the confused areas for improving global-local information fusion or aggregation. The GDBs are designed to improve segmentation performance for each point by taking into account the two-branch predictions at the 3-D point of interest and the predictions within the point's 3-D neighborhood. The GLSNet++ function is defined as

$$\mathbf{F}_{\text{GLSNet++}}(\mathbf{S}) = \mathbf{F}_{\text{Demix}}([\mathbf{S}, \mathbf{F}_{\text{Global}}(\mathbf{S}), \mathbf{F}_{\text{Local}}(\mathbf{S})]). \tag{5}$$

In GLSNet++, the proposed GDB is composed by GNN demixing layers (GDLs). In the following, we first introduce the GDL, and then present two variants of GDB aggregation.

1) GNN Demixing Layer: For each point, a demixing layer performs graph aggregation over its contextual points' features, global probability, and local probability, and the demixing block stacks up demixing layers to obtain multiple neighborhood sizes or receptive fields to improve point predictions. As shown in Fig. 4, the graph demixing layer takes a point set as input, defined as $\mathbf{S} \in N \times 1 \times \mathbb{R}^{D^{c_{\text{in}}}}$, where N is the number of points, $D^{c_{\text{in}}}$ is the dimension of an input feature vector. For the first demixing layer in each GDB, in each point p_i, its feature representation $\mathbf{s}_i$ is composed of the self LiDAR geometry feature $\{x_i, y_i, z_i\}$, LiDAR Return Intensity I_i, LiDAR Return Number R_i, and the two-branch probabilities $\mathbf{P}^i_{\text{Global}}$ and $\mathbf{P}^i_{\text{Local}}$. For the other demixing layers in the demixing block, $D^{c_{\text{in}}}$ is the number of channels of the input feature at that layer.

For each point, we construct a directed graph $G = (V, E)$ to represent the feature vectors of its neighbors, where $V = \{1, \ldots, K\}$ and $E \in V \times V$ are the vertices and edges, respectively. In the demixing layer, we construct the neighborhood graph of $\mathbf{S}$ with the grid size R, where we search for neighboring points within a radius R (i.e., [−R,+R]) from each center point along eight convenient ray directions and select $\mathcal{A}$ points per ray. The neighborhood point set is defined as $\mathbf{S}^K$ with $K = 8 * \mathcal{A}$. Note that when there are sufficient points in an octant of a grid R, we choose $\mathcal{A}$ points in the octant that are $\mathcal{A}$ nearest neighbors to the center point, and when the octant has insufficient points, we take the center point in that grid and use it as the proxy for the missing points. In the special case of $\mathcal{A} = 1$, for the octant, we either choose the nearest point to the center point or use the center point if the octant is empty. Therefore, the directed graph G for each point has K points. The graph includes self-loops, meaning that each node also points to itself. The edge features are learnable and are defined as

$$\mathbf{S}^{\text{Demixing}} = \Theta(\mathbf{S}, \mathbf{S}^K) = \mathbf{W}^{c_{\text{out}}}_{c_{\text{in}}}([\mathbf{S}, \mathbf{S}^K]) \tag{6}$$

where the nonlinear function Θ has a set of learnable parameters $\mathbf{W}^{c_{\text{out}}}_{c_{\text{in}}}$. In Fig. 3, we show how the graph demixing layer is designed to incorporate the self features, and global and local probabilities.

For each point p_i at the location $\{x_i, y_i, z_i\}$ along with its feature vector $\mathbf{s}_i$, its neighborhood set is $\mathbf{s}_i^K$. In Fig. 4, we use a point p_0 at $\{x_0, y_0, z_0\}$ with $K = 8$ (in this case $\mathcal{A} = 1$), which means that we find its eight neighbors along eight ray directions (in a $2 \times 2 \times 2$ grid). The set of neighbors for each 3-D point is denoted as $\mathbf{s}_i^K$. The input data to the graph is the set $\mathbf{S} \in N \times \mathbb{R}^{D_{c_{\text{in}}}}$, where $\mathbf{S} = \{\mathbf{s}_i\}, i = 1, 2, \ldots, N$, and the output is $\mathbf{S}^{\text{Demixing}} \in N \times \mathbb{R}^{D_{c_{\text{out}}}}$. The demixing layer concatenates the input point set $\mathbf{S}$, and the neighborhood point set $\mathbf{S}^K$, which leads to $\mathbf{S}^{\text{in}} \in N \times 9 \times \mathbb{R}^{D_{c_{\text{in}}}}$ features. A 2-D convolution layer has the parameters $\mathbf{W}^{c_{\text{out}}}_{c_{\text{in}}}$, the kernel size is 9 where each kernel in $\mathbf{W}^{c_{\text{out}}}_{c_{\text{in}}}$ is applied to the 9-D features of

input $\mathbf{S}^{\text{in}}$. This graph demixing layer design fuses the feature information of both the center point and its neighborhood points from the global and local feature embedding streams.

2) GNN Demixing Block: As shown in the blue box of Fig. 3(c), given the input features, the demixing block stacks two demixing layers with a skip link. Assuming the grid size $R = 2$, then the first demixing layer's receptive field is 8, and after the second demixing layer, the receptive field is increased to 64 (8×8). We concatenate the first demixing layer's output with the second demixing layer's output using the skip link. By doing so, each point has the receptive fields of its neighborhood in two scales, referred to as MR field GDB (abbreviated as GDB-MR). This MR scheme helps refine each point's semantic segmentation by using different levels of predictions of the surrounding points (we found this to be especially important for the building class since this class had frequent errors on boundaries).

In order to incorporate multilevel contextual and structural information, we also design an MSMR field GDB (abbreviated as GDB-MSMR), where the multiscale processing is implemented by utilizing the sampling and grouping (SG) and feature interpolation (FI) operations in Graph Demixing-MR. The SG and FI are functions similar to the downsampling and upsampling layers in the convolutional network, which are from PointNet++, where SG is a sample grouping downsampling block (i.e., choosing furthest points) and FI is a FI block.

GLSNet++ incorporates different levels of *contextual and structural information* for improved semantic label predictions by refining the global and local stream branch semantic predictions for each 3-D point. GLSNet++ with the class label GDB achieves better performance than other global and local fusion methods, including GLSNet-Parallel and GLSNet-Cascade as described in Section V-C.

IV. Experiments

A. LiDAR Datasets

1) IEEE DFT4: We first train and test the proposed GLSNet++ using the 2019 IEEE GRSS Data Fusion Contest Urban Semantic 3-D, Track 4 (3-D Point Cloud Classification Challenge) LiDAR dataset with ground truth, which we refer to as IEEE DFT4 (Data Fusion Track 4). Given a LiDAR point cloud, the task objective is to predict a semantic label for each 3-D point, which is also referred to as point cloud semantic segmentation. There are six object classes: Ground (G), High Vegetation (HV), Building (B), Water (W), Elevated Road (ER), and Unlabeled (U). The Unlabeled category was not used in the performance evaluation in IEEE DFT4. The information provided for the 3-D LiDAR point cloud consists of $\{X, Y, Z, \text{Intensity}, \text{Return Number}\}$. There are 110 point clouds, for training, 10 for validation, and another hidden 10 for testing on the server. The point clouds are from Omaha, NE and Jacksonville, FL. More details are available at [14], [54], and [55].

2) Columbia LiDAR Point Clouds (COU) for Benchmarking: We constructed an independent testing dataset consisting of LiDAR for Columbia, Missouri (COU). The high-resolution Columbia point clouds were collected by the local government of Boone County, Missouri in 2014 as a part of their regular mapping process. The data were collected by a third party using a Leica ALS70 Aerial LiDAR sensor system. The nominal collection scenario called for the acquisition of nominal point spacing of 0.7 m on the ground. To use the data to evaluate our proposed semantic segmentation networks, we cropped two large tiles of original point cloud data. The cropped point clouds are named COU-Orbit28 and COU-Orbit30. Each tile covers 1000 m $\times$ 1000 m areas. COU-Orbit28 had 4 244 969 points, and COU-Orbit30 had 4 368 678 points. In Fig. 1, we colored the COU point cloud by each point's height, intensity, and return number. Since the ground truths of the COU point cloud segmentation were unavailable, a common scenario in real application cases, we propose a new workflow to test and evaluate on these point cloud benchmark by using our system trained from a different point-cloud dataset, i.e., the IEEE DFT4 training set (to be described in Section V-B). We have released the current annotation for the two large point clouds with OSM ground-truth. Later, we will annotate more semantic classes, including Roads, Ground, Trees, and Water, as we continuously update the ground truth labels over time.

B. Implementations and Experimental Settings

GLSNet++ uses a two-stage training scheme. In the first stage, to optimize each branch, the global and local branches were trained independently. The global branch is designed to capture long-range contextual and structural information. To effectively realize this, we employed a large grid size to generate a sparse point cloud that aggregated grid features. We used a grid size of 5 m $\times$ 5 m $\times$ 1 m. If multiple points were in a voxel grid, all the input feature vectors in each grid were averaged. The SSCN-U had 16 filters in the first layer and a total of seven U-Planes, and we added 16 more filters for each U-Plane with two residual blocks. We trained the global branch with the Adam optimizer with an initial learning rate of 0.001 for 512 epochs. The local branch was designed to capture local tiles' fine contextual and structural information. Therefore, we first quantized the whole volume of a point cloud with a grid size of 1 m, and then split them into overlapping tiles. The local branch employing PointNet++ was trained with local tiles. We trained the local branch using the Adam optimizer with an initial learning rate 0.001 for 200 epochs. For additional details of data augmentation in training the global and local branches, please refer to [12]. In [12], we only used 90% training data with the remaining 10% for validation because the official validation set was not accessible when we took part in the challenge. In this work, we retrained the local branch with 100% training data for fair comparisons among different network structures and used the official validation set.

In the second stage, only the demixing block was trained to fuse the two branches' prediction results. We used the preprocessing codes provided by the baseline system of IEEE DFT4 [56] to process our data, which first quantized the point clouds by the voxel size of 1 m, and then partitioned the point clouds into tiles where each tile had around 65 536 points. We also experimented on quantization with the voxel size of 0.1 m, which performed better in our ablation study. The demixing block was trained with the training set of IEEE

TABLE I
COMPARISONS OF GLSNET++ WITH EIGHT REPRESENTATIVE ARCHITECTURES FOR URBAN POINT CLOUD SEMANTIC SEGMENTATION ON IEEE DFT4 TEST SET. IOU OF FIVE CLASSES G, HV, B, W, AND ER, ALONG WITH OA, MIOU, AND MODEL SIZE IN MEGABYTES (MB) ARE SHOWN. *PERFORMANCE AND MODEL SIZES ARE FROM [57]

Method	G	HV	B	W	ER	OA	mIoU	Model Size (MB)
PointNet++*	95.4	95.2	83.7	88.4	76.6	96.3	87.9	3.7
PointNet++(MSG)*	96.8	94.9	85.8	93.1	74.8	96.9	89.1	10.3
DGCNN*	96.4	96.2	86.3	**96.2**	48.7	96.9	84.8	9.1
PointCNN*	96.7	95.4	88.3	88.3	**83.5**	97.3	90.4	43.9
PointConv*	97.6	95.5	89.1	92.1	76.3	97.6	90.1	82.6
PointSIFT*	97.4	96.1	88.4	91.5	79.3	97.5	90.6	51.6
SSCN-U	97.4	91.8	85.5	94.2	70.0	96.9	87.8	28.6
GLSNet	97.6	93.9	87.6	95.6	77.9	97.4	90.5	32.3
GLSNet++	**97.9**	**96.2**	**90.4**	95.8	79.8	**98.0**	**92.0**	32.7

DFT4 using the Adam optimizer for 200 epochs, with a learning rate of 0.001, momentum of 0.9, and learning rate decay of 0.7. All experiments were performed with one Nvidia V100 GPU. We chose the model with the highest overall accuracy (OA) on validation data as the best model and report the best model's performance on the held-out test set on the test server from IEEE DFT4.

C. Evaluation

We used the standard measures of mean class intersection over union (mIoU) and OA for evaluation, and unlabeled points were not counted in the performance evaluation, as in [57] and [58]. Assume C semantic classes, with the classes $c = 1, 2, \ldots, C$. The mIoU was calculated by averaging the Class IoU_c

$$\text{IoU}_c = \frac{z_{cc}}{z_{cc} + \sum_{j \neq c} z_{cj} + \sum_{k \neq c} z_{kc}} \tag{7}$$

as shown in the following:

$$\text{mIoU} = \frac{\sum_{c=1}^{C} \text{IoU}_c}{C}. \tag{8}$$

Additionally, the OA is computed using

$$\text{OA} = \frac{\sum_{c=1}^{C} z_{cc}}{\sum_{j=1}^{C} \sum_{k=1}^{C} z_{jk}}. \tag{9}$$

These equations involve C semantic classes. The $C \times C$ confusion matrix, $\mathbf{z}$, was computed by the predictions of the method and ground truth, with each entry, z_{cj}, indicating the number of instances from the ground-truth class c that were predicted as class j. For evaluation on the COU point clouds, we used the workflow described below in Subsection V-B.

V. EXPERIMENTAL RESULTS

We first compare our proposed GLSNet++ with representative point cloud segmentation methods on the LiDAR IEEE DFT4 point cloud dataset. We then illustrate the generalization power of GLSNet++ to the unseen COU LiDAR point clouds and present the predicted semantic segmentation results using GLSNet++ both quantitatively and visually. Finally, ablation studies of GLSNet++ are shown for analyzing the effect of each system component.

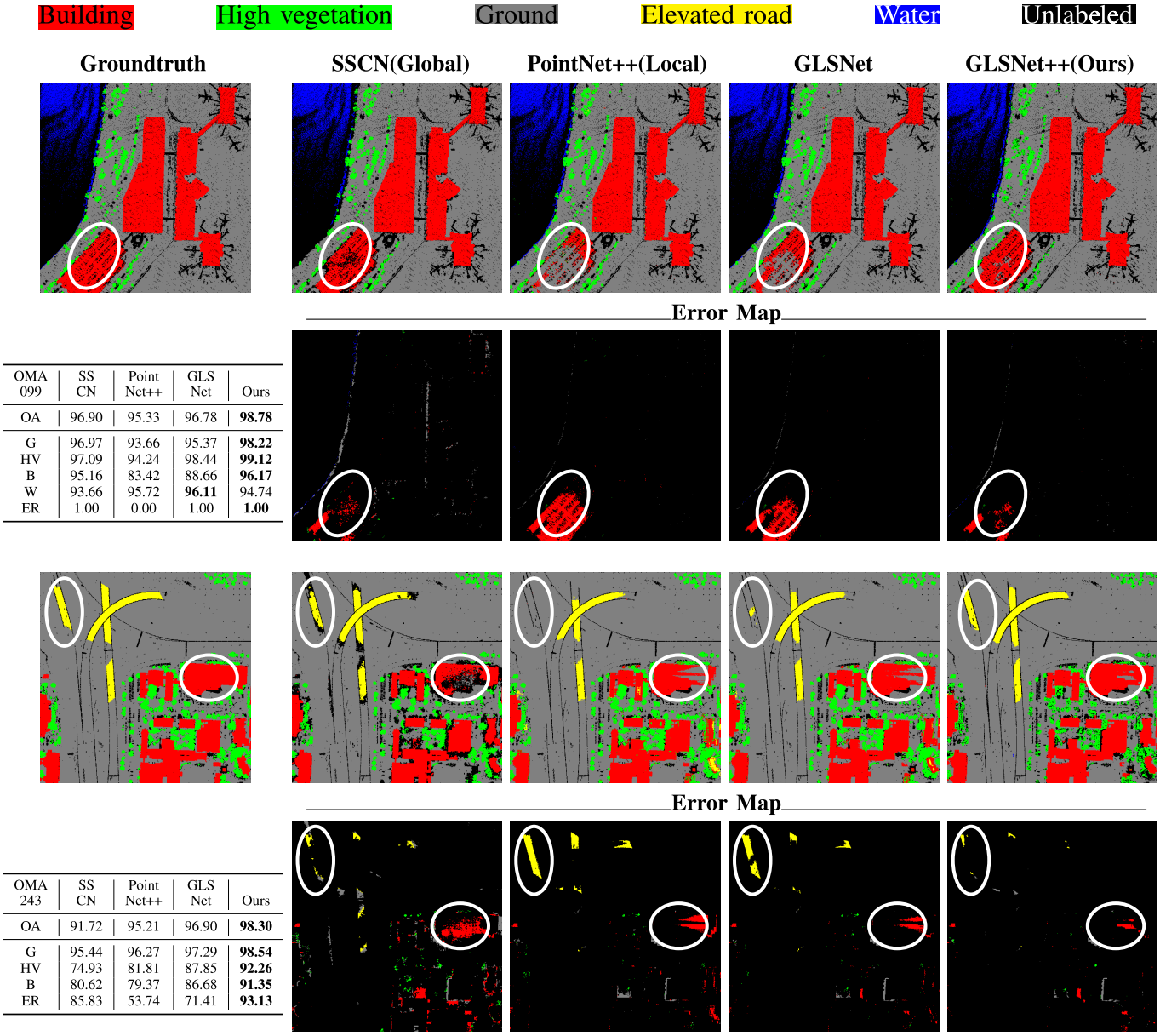
A. Comparisons With Representative Architectures

1) GLSNet++ Outperformed a Set of Representative Methods on City-Scale Point Cloud Semantic Segmentation: To compare with the representative architectures, we selected typical methods from the challenge summary paper in IEEE DFT4 [57]. Note that while these methods utilized different approaches, their fundamental structures were derived from the representative methods that we examined. It is worth mentioning that the source codes of the top-performing methods in the IEEE DFT4 challenge were not accessible to the public. Furthermore, these networks employed different preprocessing, data augmentation, and postprocessing techniques. Consequently, we conducted a comparison of our methods with the representative architectures that provide publicly available source code for point cloud segmentation on the held-out test set, as shown in Table I. Model complexity is also shown in the table. Our proposed GLSNet++ outperformed PointNet++, DGCNN, PointCNN, PointConv, PointSIFT, SSCN-U, and GLSNet, demonstrating GLSNet++'s competitive performance. GLSNet++ with demixing was overall 1.5% mIoU better than GLSNet without demixing, and it also outperformed other methods in the building, high vegetation and ground classes, demonstrating its ability to resolve complex object boundary label uncertainties using the proposed GDB.

2) Visualization of Classification Error Reduction Using Proposed GDB: To analyze the effectiveness of the GDB, we visualized the predictions of the two branches before the GDB in Table II. Errors in the global branch (second column) and in the local branch (third column) are mainly in class labels around object boundaries like buildings, high vegetation, and

TABLE II

VISUALIZATION OF 3-D POINT CLOUD SEGMENTATION METHODS FOR IEEE DFT4 LIDAR OMAHA, NE VALIDATION SET. COLUMNS SHOW GROUND TRUTH, SSCN, POINTNET++, GLSNET AND GLSNET++ FOR SAMPLE LIDAR DATASET MAPPED TO 2-D IMAGES. EVALUATION METRICS ARE SHOWN IN THE LEFTMOST COLUMN. REGIONS IDENTIFIED BY WHITE CIRCLES ILLUSTRATE GLSNET++ ADVANTAGES OVER OTHER METHODS ON CITY-SCALE POINT CLOUD SEGMENTATION, ESPECIALLY ON **B**, **HV**, AND **ER**



OMA 099	SS CN	Point Net++	GLS Net	Ours
OA	96.90	95.33	96.78	**98.78**
G	96.97	93.66	95.37	**98.22**
HV	97.09	94.24	98.44	**99.12**
B	95.16	83.42	88.66	**96.17**
W	93.66	95.72	**96.11**	94.74
ER	1.00	0.00	1.00	**1.00**

OMA 243	SS CN	Point Net++	GLS Net	Ours
OA	91.72	95.21	96.90	**98.30**
G	95.44	96.27	97.29	**98.54**
HV	74.93	81.81	87.85	**92.26**
B	80.62	79.37	86.68	**91.35**
ER	85.83	53.74	71.41	**93.13**

roads. After incorporating the GDB in GLSNet++ (last column), these class mixing errors are significantly reduced. The demixing block helps to reduce misclassification errors around object boundaries where multiple classes are close together or may overlap within a small volume region, especially for the class combinations of elevated road, building, and high vegetation. Fig. 5 shows GLSNet++ predicted results from the validation set of IEEE DFT4 for Jacksonville, FL (JAX) and Omaha, NE (OMA).

TABLE III

IOU AND OA COMPARING GLSNET TO GLSNET++ PERFORMANCE ON TWO UNSEEN COU LIDAR POINT CLOUDS FOR THE BUILDING CLASS

Method	COU-Orbit28		COU-Orbit30	
	IoU	OA	IoU	OA
GLSNet [12]	54.15	82.51	65.26	86.94
GLSNet++	**55.09**	**82.87**	**71.24**	**90.06**

B. Generalization Benchmarking Using Unseen LiDAR Point Clouds

In real scenarios, obtaining large-scale 3-D point cloud segmentation labels is often very difficult since manually labeling city-scale point clouds with millions of points is extremely time-consuming, error-prone, and difficult for expert

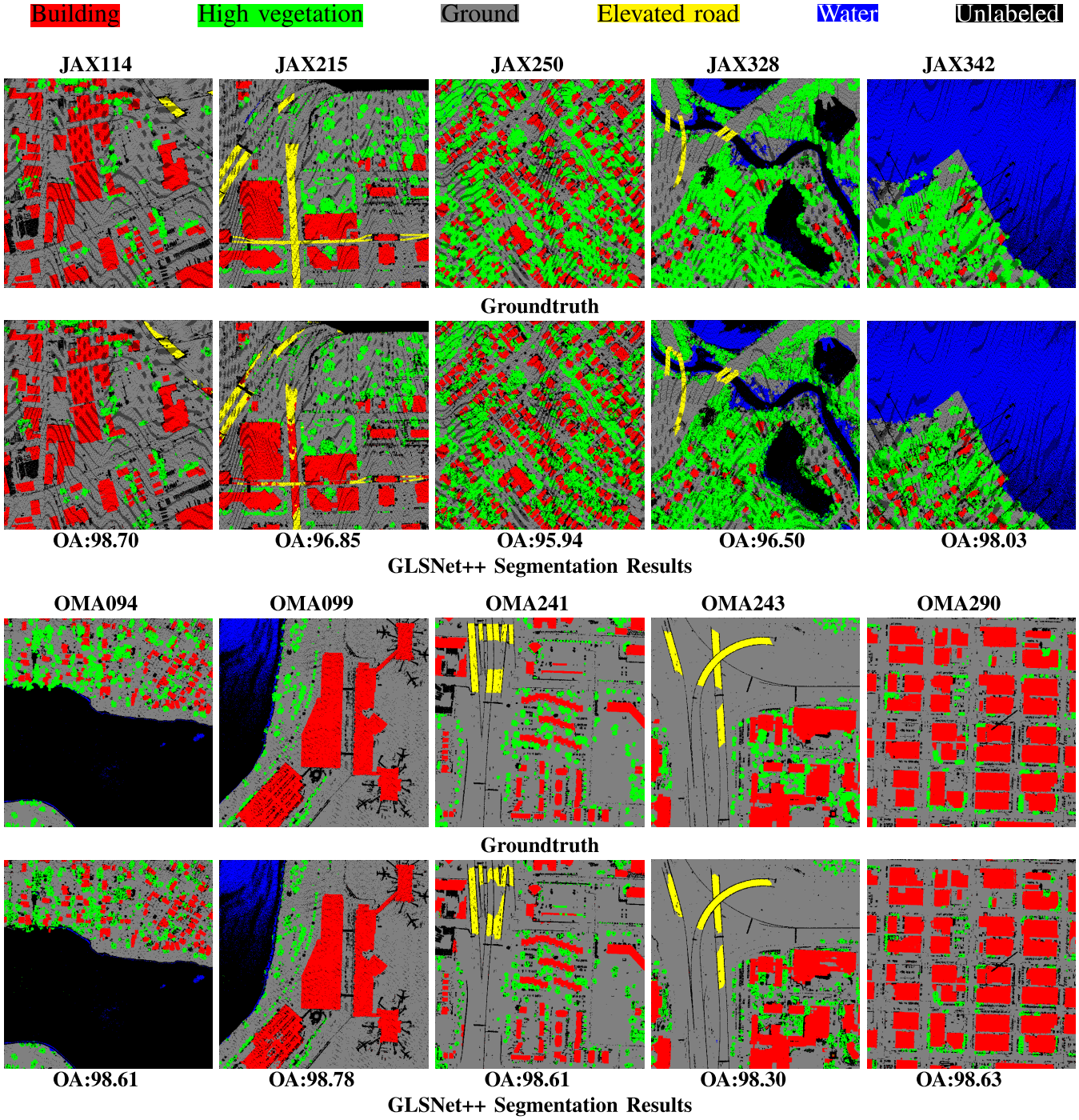


Fig. 5. Visualization of GLSNet++ on LiDAR data of validation set in the IEEE DFT4 from two cities Jacksonville, FL (JAX) and Omaha, NE (OMA). Groundtruth, and GLSNet++ for the full LiDAR dataset mapped to 2-D images are shown in each column. OA of GLSNet++ prediction is shown under each point cloud.

consensus. The flowchart in Fig. 6 shows our method to establish a benchmark for testing and evaluating on such unlabeled 3-D point clouds. Although we use the unlabeled COU LiDAR point clouds for the benchmarking, our method can be generally applied to testing on other unlabeled point cloud datasets.

1) Workflow for Evaluation on Unlabeled Point Clouds: We illustrate the workflow in Fig. 6. Before testing, we need to preprocess the unlabeled COU 3-D point clouds since the LiDAR data was a different data collection by a different vendor from that of the IEEE DFT4, which was our training set. The preprocessing procedure aligned the resolution of the testing 3-D point cloud (COU) with that of the training data (IEEE DFT4). For evaluation, OSM building footprints were used as the ground truth labels for evaluating the building segmentation performance. Note that there could be misalignments between the footprints from OSM with our collected point clouds since they were collected on different dates, and urban structures can change during this time interval.

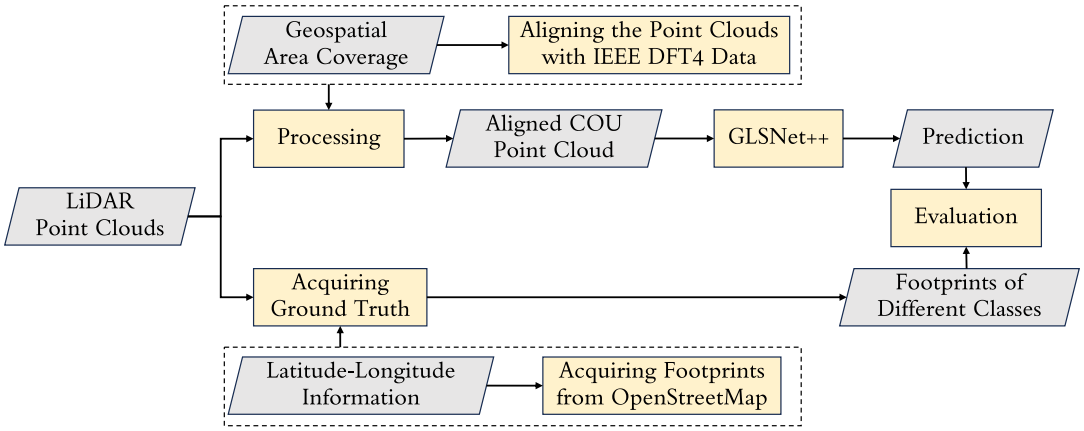


Fig. 6. Workflow for evaluating generalization capacity of GLSNet++ on unlabeled LiDAR point clouds using OSM.

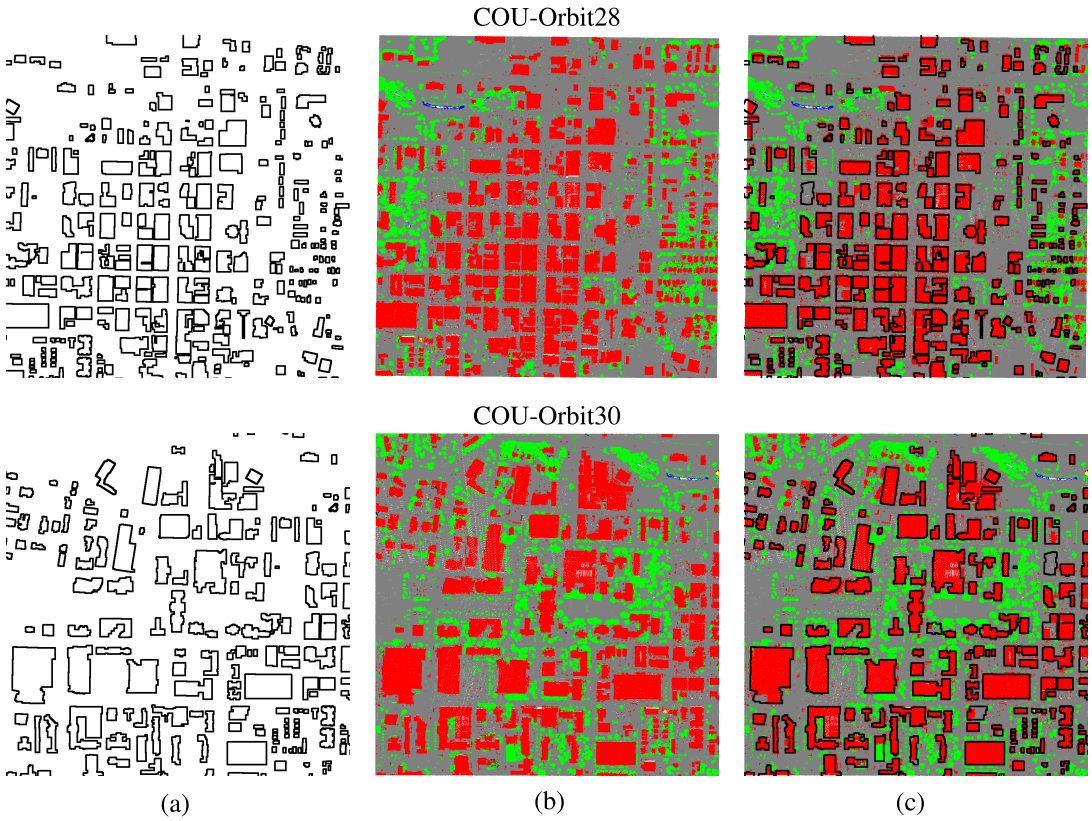


Fig. 7. Evaluation of GLSNet++ LiDAR building segmentation (red), high vegetation (green), and ground (gray) for Columbia, Missouri COU-Orbit28 (first row) and COU-Orbit30 (second row). From left to right: Building footprints from OSM, predicted class labels, and predicted results overlaid with OSM building footprints. COU-Orbit28 accuracy is 82.87% and COU-Orbit30 is 90.06% compared to OSM building footprint labels (which may have extra or missing buildings). (a) Building footprints from OSM. (b) GLSNet++ segmentation. (c) GLSNet++ segmentation with OSM building footprints.

2) Evaluation on COU LiDAR Point Clouds: We used the workflow in Fig. 6 to evaluate the performance of GLSNet++. Each tile from COU covered a 1000 m × 1000 m area, which is around four times the IEEE DFT4 data tile size, covering approximately 500 m × 500 m. COU-Orbit28 and COU-Orbit30 had over four million LiDAR points. We split the COU point clouds into four tile blocks for testing and evaluation. The quantitative results in Table III show up to a 3.1% OA generalization improvement compared to GLSNet. The building segmentation accuracy is over 80% in both COU-Orbit28 and COU-Orbit30 tiles. The visualization results are shown in Fig. 7. We observed that GLSNet++ produced very promising results for the Columbia, Missouri, LiDAR data, since our building predictions largely overlapped with the OSM building footprints. It is worth noting that GLSNet++ was trained only on the IEEE DFT4 training set (point clouds for Jacksonville, Florida, and Omaha, Nebraska), and the only additional information that we used in performing semantic point cloud segmentation for the Columbia, Missouri, data was a scale factor (the latitude and longitude of point cloud

TABLE IV
COMPARISONS OF SINGLE-STREAM, DUAL-STREAM, AND GLSNET++ ENSEMBLES ON IEEE DFT4. BEST PERFORMING GLSNET++-MSMR COMBINES LOCAL- AND GLOBAL-BACKBONES FOLLOWED BY MSMR GDB

Streams	Method	Parameter Size	G	HV	B	W	ER	OA	mIoU
Single-Stream	Global-branch	7.531M	97.40	91.82	85.50	94.18	69.98	96.88	87.77
	Local-branch	0.970M	96.73	95.12	86.19	93.21	76.08	96.41	89.47
Dual-Stream	GLSNet-Parallel	8.501M	97.73	93.83	88.02	96.44	76.47	97.47	90.50
	GLSNet-Cascade	8.501M	97.34	95.05	88.57	93.23	78.17	97.48	90.47
GLSNet++	GLSNet++-MR	8.568M	97.83	96.06	90.15	95.65	79.81	97.92	91.90
	GLSNet++-MSMR	8.634M	97.86	96.20	90.37	95.84	79.84	**97.97**	**92.02**

TABLE V
PARAMETERS OF PROPOSED GDL AND GDB ARCHITECTURES

Module	Layer	Kernel size	# of input channels	# of output channels	# of Param.	Total Param.
GDL	Conv	9×1	c_{in}	c_{out}	$9 \times 1 \times c_{in} \times c_{out} + c_{out}$	
	BN	–	–	–	$2 \times c_{out}$	
GDB-MR	DL1	9×1	17	64	$9 \times 1 \times 17 \times 64 + 64 + 2 \times 64$	66,310
	DL2	9×1	64+3	64	$9 \times 1 \times 67 \times 64 + 64 + 2 \times 64$	
	Conv	1×1	128	128	$1 \times 1 \times 128 \times 128 + 128$	
	BN	–	–	–	128+128	
	Classifier	1×1	128	6	$1 \times 1 \times 128 \times 6 + 6$	
GDB-MSMR	DL1-s1	9×1	17	64	$9 \times 1 \times 17 \times 64 + 64 + 2 \times 64$	133,190
	DL2-s1	9×1	64+3	64	$9 \times 1 \times 67 \times 64 + 64 + 2 \times 64$	
	DL1-s2	9×1	17	64	$9 \times 1 \times 17 \times 64 + 64 + 2 \times 64$	
	DL2-s2	9×1	64+3	64	$9 \times 1 \times 67 \times 64 + 64 + 2 \times 64$	
	Conv	1×1	256	128	$1 \times 1 \times 256 \times 128 + 128$	
	BN	–	–	–	128+128	
	Classifier	1×1	128	6	$1 \times 1 \times 128 \times 6 + 6$	

tiles) for calibrating the resolution of the test 3-D point clouds relative to the training point clouds.

C. Ablation Study

We performed a set of ablation studies on GLSNet++ to evaluate the factors of stream design, fusion design, and hyperparameters.

1) Dual-Stream Networks Outperform Single-Stream Networks: In comparison with the Single-Stream approach, our Dual-Stream networks outperformed the single-stream networks up to 2.73% mIoU (GLSNet-Parallel versus Global-Branch), with more details in Table IV. While GLSNet-Cascade and GLSNet-Parallel exhibited similar performance in terms of mIoU and OA, there were discernible variations in accuracy for individual classes. The marginal improvement in OA over each branch was attributed to the straightforward max pooling fusion employed in GLSNet-Parallel or the simple local refinement mechanism employed in GLSNet-Cascade. Our proposed GLSNet++ with the MSMR exhibits a remarkable superiority over both GLSNet-Parallel (+1.52% mIoU) and GLSNet-Cascade (+1.55% mIoU).

2) GLSNet++ With Demixing Block Is Effective and Efficient: Table V provides details of the GDL, GDB-MR, and GDB-MSMR architectures and their parameter sizes. In a demixing block, the demixing layer is composed of a convolution layer (Conv) with the number of input channel size c_{in}, the number of output channel size c_{out}, the kernel size 9×1 (parameter size $9 \times 1 \times c_{\text{in}} \times c_{\text{out}} + c_{\text{out}}$), and a batch normalization layer (BN) (parameter size c_{out} for beta and parameter size c_{out} for gamma in BN). In GDB-MR and GDB-MSMR, DL2 has a +3 in the number of input channels because we concatenated the $\{X, Y, Z\}$ information in the layer to encode the geometry information. In Table V, GDB-MR and GDB-MSMR are summarized to have around 0.067M and 0.133M parameters, respectively, which are very lightweight.

3) Effectiveness and Efficiency of GDB: We compared the proposed graph demixing block with different designs of global-local fusion in GLSNet++, as given in Table VI. We chose PointNet++ and DeepGCN [40], [41] as fusion blocks in the comparison because they are representative methods in GNNs, and our demixing block GDB is a graph-based dual scale information fusion method. Specifically, PointNet++ is a widely used point graph network. GNNs incorporate the advantages of CNNs such as residual connections, dense connections, and dilated convolutions. In Table VI, we show the drop in performance by replacing our demixing block with PointNet++ or DeepGCN to fuse

TABLE VI
COMPARISONS OF USING OUR GDB FOR AGGREGATION VERSUS POINTNET++ OR DEEPGCN [41] FOR FUSING TWO STREAMS ON IEEE DFT4 LIDAR. OUR PROPOSED GDB ACHIEVES BETTER RESULTS WITH FEWER PARAMETERS

Fusion Method	Parameter Size	G	HV	B	W	ER	OA	mIoU
PointNet++	0.970M	97.29	95.29	88.36	95.70	80.90	97.50	91.51
DeepGCN-N8-D2	0.869M	97.38	94.87	88.12	96.13	79.60	97.48	91.22
DeepGCN-N8-D7	1.402M	97.21	94.76	87.81	95.58	75.16	97.34	90.10
Our GDB-MR	**0.067M**	97.83	96.06	90.15	95.65	79.81	**97.92**	**91.90**
Our GDB-MSMR	**0.133M**	97.86	96.20	90.37	95.84	79.84	**97.97**	**92.02**

TABLE VII
ABLATION STUDY FOR DIFFERENT VOXELIZATION RESOLUTIONS AND SAMPLING POINT SET SIZES IN GDB

Method	Voxel Size (m)	Sampling Size	G	HV	B	W	ER	OA	mIoU
GLSNet++-MR	1	8,192	97.39	95.57	88.75	96.02	79.41	97.60	91.43
GLSNet++-MR	0.1	8,192	97.62	95.85	89.56	95.31	80.95	97.77	91.86
GLSNet++-MR	0.1	16,384	97.83	96.06	90.15	95.65	79.81	**97.92**	**91.90**

the two branches, where GCN-N8 represents using eight neighbors, and D2 or D7 uses two layers or seven layers in DeepGCN. It is observed that by using a simple demixing block (GDB-MR), i.e., GLSNet++-MR, we can achieve performance comparable to PointNet++ and DeepGCN, indicating that our GDB is very effective for fusing the global-local branches. GLSNet++-MR, with its lightweight demixing block, achieved an mIoU performance of 91.90% and GLSNet++-MSMR achieved an mIoU performance of 92.02%. The total parameter size of GDB-MR is around 0.067M, which is about 7% of the parameter size of PointNet++, and 8% of the parameter size of DeepGCN-N8-D2. The total parameter size of our GDB-MSMR is around 0.133M which is only 14% of the parameter size of PointNet++, and 15% of DeepGCN-N8-D2. Even with a much lighter and compact design, our GLSNet++ with GDB-MSMR still outperforms the PointNet++-based fusion scheme by 1.51%, and outperforms DeepGCN-N8-D7 by 1.92% in mIoU, demonstrating its efficiency.

4) Hyperparameter Analysis for GDB: We explored different values for two hyperparameters in designing the demixing block: voxelization resolution and sampling size. The results are given in Table VII. *Voxel size:* Voxel size is the cell size used in the voxelization procedure for a demixing block. For instance, if we use a voxel size of 1 m, then the voxelized cell will be quantized with 1 m before the whole point cloud is split into tiles, as discussed in the implementation of Section IV. In our ablation study, we tried voxel size of 1 and 0.1 m. Comparing the first row and the second row in the table, we can see that using 0.1 m gave a better performance than using 1 m, since using 0.1 m can almost keep the original point cloud resolution of IEEE DFT4 with its LiDAR density of 8 cm. In such a setting, GLSNet++-MR can use finer details to demix the error areas. *Sampling size:* Sampling size is the input size for the network. Tile prediction is voted by the prediction of the sampled points. Increasing the input point size in a demixing block means that for each tile, we use more points to estimate the tile's semantic segmentation results since the results are interpolated from the preprocessed sets of points. We can see that doubling the sampling size to 16 384 (third row) achieved better results than a sampling size of 8192 (second row) because the network was provided with more points in training and inference.

VI. CONCLUSION

Accurate city-scale classification of 3-D LiDAR point clouds based solely on range measurements can be achieved using voxel- and point-based deep learning architectures. Our work has demonstrated that the proposed GLSNet++ architecture, incorporating dual global and local feature ensemble streams, surpassed single-stream architectures by up to 2.7% in mIoU. By leveraging a lightweight GDB to fuse the global and local feature embeddings and to refine mixed class labels near object boundaries, further improves accuracy by approximately 1.5% mIoU. As a result, we have achieved an mIoU of 92% across five common urban classes, even in the presence of significant class imbalance and sparse training data. The performance of GLSNet++ with feature fusion achieved competitive performance with a set of representative methods. Furthermore, we have proposed a benchmark and evaluation workflow using OSM when point cloud labels are unavailable to assess the generalization power of GLSNet++. Our GDB spatial contextual and structural MSMR field feature fusion or aggregation approach in GLSNet++ significantly enhances generalization to aerial LiDAR data for new unseen urban scenes and demonstrates its potential for large-scale smart city modeling, digital twin simulation, and navigation applications.

ACKNOWLEDGMENT

The COU LiDAR data was provided by Nathan Mattox, geographical information system (GIS) Manager, Boone County Government, Missouri IT Department. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the U. S. Government or agency thereof.

REFERENCES

[1] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep learning for 3D point clouds: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 12, pp. 4338–4364, Dec. 2021.

[2] S. Ortega, J. M. Santana, J. Wendel, A. Trujillo, and S. M. Murshed, "Generating 3D city models from open LiDAR point clouds: Advancing towards smart city applications," in *Proc. Open Source Geospatial Sci. Urban Stud. (LNITI)*, 2021, pp. 97–116.

[3] H. AliAkbarpour, K. Palaniappan, and G. Seetharaman, "Parallax-tolerant aerial image georegistration and efficient camera pose refinement—Without piecewise homographies," *IEEE Trans. Geosci. Remote Sens.*, vol. 55, no. 8, pp. 4618–4637, Aug. 2017.

[4] H. Aliakbarpour, K. Palaniappan, and G. Seetharaman, "Robust camera pose refinement and rapid SfM for multiview aerial imagery—Without RANSAC," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 11, pp. 2203–2207, Nov. 2015.

[5] Y. Cui et al., "Deep learning for image and point cloud fusion in autonomous driving: A review," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 2, pp. 722–739, Feb. 2021.

[6] P. R. W. Urech, M. A. Dissegna, C. Girot, and A. Grêt-Regamey, "Point cloud modeling as a bridge between landscape design and planning," *Landscape Urban Planning*, vol. 203, Nov. 2020, Art. no. 103903.

[7] R. Wang, J. Peethambaran, and D. Chen, "LiDAR point clouds to 3-D urban models: A review," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 2, pp. 606–627, Feb. 2018.

[8] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial], Part I: The first 30 years and fundamentals," *IEEE Robot. Autom. Mag.*, vol. 18, no. 4, pp. 80–92, 2011.

[9] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part II: Matching, robustness, optimization, and applications," *IEEE Robot. Autom. Mag.*, vol. 19, no. 2, pp. 78–90, Jun. 2012.

[10] H. AliAkbarpour, K. Gao, R. Aktar, S. Suddarth, and K. Palaniappan, *Structure-from-motion and Mosaicking for High-throughput Field-scale Phenotyping*. New York, NY, USA: Springer, 2021, ch. 2, pp. 55–69.

[11] J. Behley et al., "SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9297–9307.

[12] R. Bao, K. Palaniappan, Y. Zhao, G. Seetharaman, and W. Zeng, "GLSNet: Global and local streams network for 3D point cloud classification," in *Proc. IEEE Appl. Imag. Pattern Recognit. Workshop (AIPR)*, Oct. 2019, pp. 1–9.

[13] Z. Wu et al., "3D ShapeNets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1912–1920.

[14] *2019 IEEE GRSS Data Fusion Contest*. Accessed: 2019. [Online]. Available: http://www.classic.grss-ieee.org/community/technicalcommittees/data-fusion/2019-ieee-grss-data-fusion-contest/

[15] Y. Zhang, Y. Yang, X. Gao, L. Xu, B. Liu, and X. Liang, "Robust extraction of multiple-type support positioning devices in the catenary system of railway dataset based on MLS point clouds," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023, Art. no. 5702314.

[16] 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 CVPR*, Jul. 2017, pp. 652–660.

[17] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 5099–5108.

[18] M. Atzmon, H. Maron, and Y. Lipman, "Point convolutional neural networks by extension operators," 2018, *arXiv:1803.10091*.

[19] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution on *X*-transformed points," in *Proc. NeurIPS*, 2018, pp. 820–830.

[20] T. Le and Y. Duan, "PointGrid: A deep network for 3D shape understanding," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9204–9214.

[21] B.-S. Hua, M.-K. Tran, and S.-K. Yeung, "Pointwise convolutional neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 984–993.

[22] H. Zhao, L. Jiang, C.-W. Fu, and J. Jia, "PointWeb: Enhancing local neighborhood features for point cloud processing," in *Proc. CVPR*, Jun. 2019, pp. 5565–5573.

[23] M. Zhang, H. You, P. Kadam, S. Liu, and C.-C. J. Kuo, "PointHop: An explainable machine learning method for point cloud classification," *IEEE Trans. Multimedia*, vol. 22, no. 7, pp. 1744–1755, Jul. 2020.

[24] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, and C. Lu, "PointSIFT: A SIFT-like network module for 3D point cloud semantic segmentation," 2018, *arXiv:1807.00652*.

[25] X. Liu, Z. Han, Y.-S. Liu, and M. Zwicker, "Point2sequence: Learning the shape representation of 3D point clouds with an attention-based sequence to sequence network," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, Jul. 2019, pp. 8778–8785.

[26] Q. Hu et al., "RandLA-Net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11108–11117.

[27] C. Choy, J. Gwak, and S. Savarese, "4D spatio-temporal convnets: Minkowski convolutional neural networks," in *Proc. IEEE CVPR*, 2019, pp. 3075–3084.

[28] F. Zhang et al., "Instance segmentation of LiDAR point clouds," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 9448–9455.

[29] B. Graham and L. van der Maaten, "Submanifold sparse convolutional networks," 2017, *arXiv:1706.01307*.

[30] J. Hou, A. Dai, and M. Niessner, "3D-SIS: 3D semantic instance segmentation of RGB-D scans," in *Proc. IEEE CVPR*, 2019, pp. 1–10.

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

[32] C. R. Qi, H. Su, M. Nießner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5648–5656.

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

[34] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, "Semantic3D.Net: A new large-scale point cloud classification benchmark," 2017, *arXiv:1704.03847*.

[35] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-net: Learning dense volumetric segmentation from sparse annotation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Interv.*, 2016, pp. 424–432.

[36] B. Graham, M. Engelcke, and L. van der Maaten, "3D semantic segmentation with submanifold sparse convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 9224–9232.

[37] F. Groh, P. Wieschollek, and H. P. A. Lensch, "Flex-convolution: Million-scale point-cloud learning beyond gridworlds," in *Proc. Asian Conf. Comput. Vis.* Cham, Switzerland: Springer, 2018.

[38] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "SpiderCNN: Deep learning on point sets with parameterized convolutional filters," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 87–102.

[39] L. Wang, Y. Huang, Y. Hou, S. Zhang, and J. Shan, "Graph attention convolution for point cloud semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10296–10305.

[40] G. Li, M. Müller, A. Thabet, and B. Ghanem, "DeepGCNs: Can GCNs go as deep as CNNs?" in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9266–9275.

[41] G. Li et al., "DeepGCNs: Making GCNs go as deep as CNNs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 6, pp. 6923–6939, Jun. 2023.

[42] Q. Huang, W. Wang, and U. Neumann, "Recurrent slice networks for 3D segmentation of point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2626–2635.

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

[44] L. Landrieu and M. Simonovsky, "Large-scale point cloud semantic segmentation with superpoint graphs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4558–4567.

[45] F. Yu, K. Liu, Y. Zhang, C. Zhu, and K. Xu, "PartNet: A recursive part decomposition network for fine-grained and hierarchical shape segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9483–9492.

[46] T. Xu, X. Gao, Y. Yang, L. Xu, J. Xu, and Y. Wang, "Construction of a semantic segmentation network for the overhead catenary system point cloud based on multi-scale feature fusion," *Remote Sens.*, vol. 14, no. 12, p. 2768, Jun. 2022.

[47] Y. Lian, T. Feng, and J. Zhou, "A dense PointNet++ architecture for 3D point cloud semantic segmentation," in *Proc. IEEE IGARSS*, 2019, pp. 5061–5064.
[48] M. Jia, A. Li, and Z. Wu, "A global point-sift attention network for 3D point cloud semantic segmentation," in *Proc. IGARSS IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2019, pp. 5065–5068.
[49] 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 *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 1887–1893.
[50] 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. (CVPR)*, Jun. 2020, pp. 9601–9610.
[51] F. Zhang, J. Fang, B. Wah, and P. Torr, "Deep FusionNet for point cloud semantic segmentation," in *Proc. ECCV*, 2020, pp. 644–663.
[52] X. Wang, J. He, and L. Ma, "Exploiting local and global structure for point cloud semantic segmentation with contextual point representations," in *Proc. NeurIPS*, 2019, pp. 4571–4581.
[53] C. Liu et al., "Context-aware network for semantic segmentation toward large-scale point clouds in urban environments," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5703915.
[54] M. Bosch, K. Foster, G. Christie, S. Wang, G. D. Hager, and M. Brown, "Semantic stereo for incidental satellite images," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2019, pp. 1524–1532.
[55] B. Le Saux, N. Yokoya, R. Hansch, M. Brown, and G. Hager, "2019 data fusion contest [technical committees]," *IEEE Geosci. Remote Sens. Mag. Replaces Newsletter*, vol. 7, no. 1, pp. 103–105, Mar. 2019.
[56] *2019 IEEE GRSS Data Fusion Contest Baseline*. Accessed: 2019. [Online]. Available: https://github.com/pubgeo/dfc2019/tree/master/track4/
[57] Y. Lian et al., "Large-scale semantic 3-D reconstruction: Outcome of the 2019 IEEE GRSS data fusion contest—Part b," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 1158–1170, 2021.
[58] T. Hackel, N. Savinov, L. Ladicky, J. D. Wegner, K. Schindler, and M. Pollefeys, "Semantic3D.Net: A new large-scale point cloud classification benchmark," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 1, pp. 91–98, May 2017.

Rina Bao received the B.E. degree from Tianjin University, Tianjin, China, in 2014, and the Ph.D. degree from the University of Missouri, Columbia, MO, USA, in 2021.

She is currently a Postdoctoral Research Fellow at Boston Children's Hospital and Harvard Medical School, Boston, MA, USA. Her research interests include data understanding with machine learning and deep learning methods. She mainly focuses on medical data analysis now.

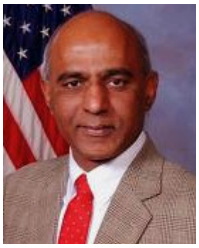
Kannappan Palaniappan (Senior Member, IEEE) received the Ph.D. degree from the University of Illinois at Urbana–Champaign, Champaign, IL, USA, in 1991.

He is a Curators' Distinguished Professor at the Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO, USA. His research interests include computer vision, high performance computing, data science, biomedical image analysis, and remote sensing.

Dr. Palaniappan was an Associate Editor of IEEE TRANSACTIONS ON IMAGE PROCESSING.

Yunxin Zhao (Life Senior Member, IEEE) received the B.S. degree from the Beijing Institute of Posts and Telecommunications, Beijing, China, in 1982, and the M.S.E.E. and Ph.D. degrees from the University of Washington, Seattle, WA, USA, in 1985 and 1988, respectively.

She is a Professor at the Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO, USA. Her research focuses on spoken language processing, speech and signal processing, machine learning, and biomedical applications.



Guna Seetharaman (Fellow, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Miami, Coral Gables, FL, USA, in 1988.

He was a Principal Engineer of Computing Architectures with the Information Directorate, Air Force Research Laboratory (AFRL), Rome, NY, USA, until 2014. He is currently a Navy Senior Scientist (ST) for Advanced Computing Concepts, and the Chief Scientist at the Information Technology Division, U.S. Naval Research Laboratory (NRL), Washington, DC, USA. His research interests include high performance computing, computer vision, and information exploitation.