Automation in Construction 181 (2026) 106631

Contents lists available at ScienceDirect

Automation in Construction

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

Domain-generalizable point cloud instance segmentation of bridge components using class-balanced dynamic thresholding

Jiawei Xu [a], Mingyu Shi [a,b], Rafael Cabral [c], Diogo Ribeiro [c,d], Weilei Yu [e], Huayong Wu [f], Yasutaka Narazaki [a,*]

[a] Zhejiang University-University of Illinois Urbana-Champaign Institute, Zhejiang University, Haining, Zhejiang, China
[b] College of Civil Engineering and Architecture, Zhejiang University, Hangzhou, Zhejiang, China
[c] CONSTRUCT - LESE, Faculty of Engineering, University of Porto, Porto, Portugal
[d] iBuilt, Instituto Superior de Engenharia do Porto, Polytechnic of Porto, Porto, Portugal
[e] Department of Engineering Mechanics and Energy, University of Tsukuba, Tsukuba, Ibaraki, Japan
[f] Shanghai Key Laboratory of Engineering Structure Safety, Shanghai Research Institute of Building Science and Technology, Shanghai, China

ARTICLE INFO

Keywords:
Bridge point cloud data
Instance segmentation
Unsupervised domain adaptation
Self-training
Synthetic data
Deep learning

ABSTRACT

Segmenting bridge point cloud data into component instances is challenging due to noise and class imbalance in the field data, as well as distribution shifts from training data. To address these issues, this paper proposes an approach comprising: (1) an automated synthetic data generation covering multiple bridge types, (2) an unsupervised domain adaptation framework for 3D instance segmentation, and (3) a class-balanced dynamic thresholding method for robust pseudo-labeling. Extensive experiments were conducted on four real-world domains, using one domain as the unlabeled target domain during training and the others as unseen domains for evaluation. The proposed method achieved a + 2.5 % improvement in average precision (AP) on the target domain and up to +11.1 % improvement in AP on the unseen domains, compared to the synthetic-only baseline. These results highlight the effectiveness of the proposed approach in achieving high-precision and domain-generalizable component extraction, supporting automated bridge Scan-to-Building Information Modeling processes.

1. Introduction

Effective management of bridges is crucial for connecting communities and promoting economic growth. Traditional approaches to bridge management involve manual, inefficient, and costly steps, such as visually inspecting critical components, measuring key dimensions, and ensuring consistency with design drawings [1,2]. The digitization of bridge management presents a promising approach to enhancing the effectiveness and efficiency of these processes. In this context, digital twins (DTs) are often developed to represent the condition of their physical counterparts [3]. Scan-to-Bridge Information Modeling (Scan-to-BrIM) techniques offer a method for obtaining such DT models, using accurate geometric data from reality-capture models (typically point cloud data) of existing assets [4]. Scan-to-BrIM generally consists of two main steps: (1) the detection of bridge components from point clouds and (2) the creation of geometric models by fitting shapes and their spatial relationships (e.g., based on the Industry Foundation Classes (IFC) [5]). By establishing an efficient and automated pipeline to identify bridge components from point clouds, the scan-to-BrIM and digital twinning processes can be significantly improved.

Existing methods for segmenting bridge point clouds can be broadly categorized into heuristic-based and deep learning (DL)-based approaches. While heuristic-based methods demonstrate utility for extracting components for semantic-level and instance-level information from relatively simple bridge types [6–9], their reliance on a priori geometric assumptions, manual parametric thresholds, and rigid topological rules constrains their effectiveness when applied to complex or non-standard bridge structures. DL-based segmentation algorithms offer potential for directly segmenting bridge point clouds by automatically extracting structural features. DL-based semantic segmentation has demonstrated promising results in enhancing Scan-to-BrIM processes [10,11]. The extension of such work to instance-level segmentation of bridge components can be supported by recent algorithmic developments, including top-down, proposal-based methods [12–14] and

* Corresponding author.
E-mail address: narazaki@intl.zju.edu.cn (Y. Narazaki).

https://doi.org/10.1016/j.autcon.2025.106631
Received 12 May 2025; Received in revised form 8 October 2025; Accepted 24 October 2025
Available online 4 November 2025

bottom-up, proposal-free methods [15–22]. However, training DL models for bridge instance segmentation requires large, diverse datasets with detailed instance-level labels, which are costly and often inaccessible for real-world point clouds of complex bridges.

To alleviate data scarcity, synthetic data generation has been explored for bridge components. Existing research primarily focuses on parametric modeling of single bridge types with mesh sampling techniques [11,23,24], resulting in uniformly distributed point clouds that differ significantly from real-world scans. While few studies have investigated LiDAR-based [25] or SfM-based [26] simulation methods, existing synthetic datasets do not fully leverage instance-level labels for diverse bridge types.

The domain gap between synthetic and real-world data significantly impacts model performance. Synthetic datasets inevitably simplify complex real-world environments, as not all variations, terrain conditions, or surface textures can be modeled. Unsupervised domain adaptation (UDA) [27–29] offers potential to bridge this domain shift. While UDA has achieved success in 2D segmentation tasks [30–32], including bridge-related problems [33], its application to 3D point cloud segmentation remains limited [34]. Applying UDA to instance segmentation of bridge point clouds is particularly challenging due to severe class imbalance and significant distribution shifts.

This research proposes a UDA approach for DL-based bridge point cloud instance segmentation under severe class imbalance and distribution shifts. The proposed approach leverages automated synthetic data generation and a ST method termed class-balanced dynamic thresholding (CBDT). Specifically, the contributions of this paper are summarized as follows:

- Building on our preliminary research about automated generation of synthetic models of six types of bridges [35] and the associated point cloud semantic segmentation datasets [26], this research develops a data generation pipeline for diverse Structure-from-Motion (SfM) bridge point clouds with ground truth instance labels.
- Establishing a UDA approach for 3D DL-based instance segmentation that leverages the generated synthetic data and unlabeled real-world bridge point clouds, enabling performance improvements in real-world environments without manual annotations.
- Developing a ST method, termed CBDT, to deal with severe class imbalance and distribution shifts characteristic of bridge point cloud instance segmentation tasks. By analyzing the confidence probability distributions for each class at every iteration, CBDT dynamically updates thresholds for pseudo-labeling in a data-driven manner, significantly reducing reliance on predetermined rules whose optimal settings are problem-specific.

  Moreover, the proposed approach is evaluated on real-world bridge point cloud datasets from four domains, which differ in data acquisition methods, bridge types, structural dimensions, and geographic locations.

The remainder of this paper is organized as follows. Section 2 discusses the related work and research objectives. Section 3 presents the proposed approach, including the synthetic data generation pipeline, the UDA-based instance segmentation framework, and the CBDT method. Section 4 describes the experimental settings and evaluation schemes. Section 5 analyses the experimental results and discusses their practical implications. Section 6 concludes the paper and discusses future work.

2. Related work

This section reviews the state-of-the-art in instance segmentation, synthetic data generation, and unsupervised domain adaptation for bridge point clouds, and identifies the research gap and objective.

2.1. Bridge point cloud instance segmentation

Instance segmentation provides element-level information for each point beyond semantic segmentation, where all components of the same type are treated uniformly. Lamas et al. [6] employed a heuristic method for instance segmentation of truss bridge point clouds. The algorithm first partitions the truss into horizontal and vertical planes, then combines principal component analysis (PCA) and clustering algorithms to analyze the neighborhood distribution of each point. Each truss element is subsequently segmented and classified into six component classes. Although this research demonstrates the feasibility of instance segmentation for truss bridges, the detected components were only partially segmented. Also, these heuristic-based methods generally rely on a priori geometric assumptions, manually set parametric thresholds, and hard-coded topological rules, which limit their applicability to complex, non-standard, and/or noisy bridge point cloud data.

DL-based point cloud instance segmentation algorithms have been investigated actively in related domains. Methods include top-down, proposal-based methods [12–14] and bottom-up, proposal-free methods, with the latter further divided into grouping-based [16–19] and transformer-based [15,20–22] methods. SoftGroup++ [18,19] has a two-stage pipeline, benefiting from common proposal-based and grouping-based approaches. In the bottom-up stage, high-quality object proposals are produced by soft semantic score grouping, and in the top-down stage, proposals are refined to generate the final instance predictions.

Applications of these DL-based instance segmentation methods to bridge point cloud segmentation problems have been explored as robust and generalizable alternatives to heuristic-based methods. Lamas et al. [24] first employed a DL-based instance segmentation network, termed JSNet, to simultaneously obtain semantic and instance segmentation of truss bridge point clouds, achieving higher accuracy on synthetic data than previous heuristic methods, although lacking validation on real-world data. Rahman et al. [25] applied Mask3D, a transformer-based instance segmentation algorithm, on beam bridges. However, both frameworks have been designed and applied to a single bridge type. Extending DL-based instance segmentation to real-world bridge point clouds with diverse bridge types requires extensive, large-scale labeled datasets, which are costly and time-consuming to obtain, and are less commonly shared publicly than other general datasets (e.g., for general indoor scenes).

2.2. Synthetic data generation

Synthetic point cloud data has been leveraged to augment training data for bridge component segmentation [36]. Yang et al. [11], Jing et al. [23], and Lamas et al. [24] sampled points from the mesh surface to generate synthetic point clouds of reinforced concrete (RC) beam bridges, masonry arch bridges, and truss bridges, respectively, with semantic labels assigned to structural components and occlusion effects simulated by removing parts of the point clouds. Although such synthetic datasets have been successfully used to train DL-based models, sampling points from mesh surfaces produces uniformly distributed point clouds without fully accounting for real-world noise and imperfections, and some approaches still rely on labor-intensive manual labeling.

To better represent the point clouds captured in the field, Rahman et al. [25] generated synthetic point clouds of RC beam bridges using simulated LiDAR. 3D bridge geometry was modeled with randomly defined cross-sections of structural components. Point clouds with both semantic and instance labels were produced by strategically placing simulated LiDAR sensors around bridge models. Additionally, a sparsity-based occlusion algorithm was employed for data augmentation. Shi et al. [26] proposed a method to automatically generate SfM-based point cloud semantic segmentation datasets for six types of bridges. Point clouds were generated by simulating UAV-based image collection

scenarios in photo-realistic synthetic environments, and then applying the SfM algorithm to the collected synthetic images. Point clouds generated in this approach closely represent the actual SfM point clouds, containing non-Gaussian noises and local imperfections caused by non-ideal viewpoints (occlusion, far distance, limited image overlaps, etc.), surface textures, and lighting conditions. The produced datasets have been shown to improve the semantic segmentation accuracy compared to the clean and ideal point cloud datasets generated by sampling points from mesh surfaces.

A major challenge of algorithm training using synthetic bridge point cloud datasets is that the effectiveness is strictly bound to the representational quality of the synthetic environments. Many existing works specifically target a certain (or a certain type of) bridge(s), and design synthetic data generation pipelines to represent the target application scenarios accurately. Moreover, significant simplifications are imposed during the synthetic model preparation process to enable efficient data generation, omitting many geometric and textural details and diversity. Even after preparing the synthetic bridge models, additional simplifications in data collection schemes are applied, making the data clean, ideal, and therefore less realistic (despite the partial attempts to overcome this challenge [25,26]). Removing these simplifications by manual modeling requires a prohibitive amount of effort for bridge point cloud instance segmentation problems, which are less well-established than general problems (e.g., segmentation of chairs, cars, and pedestrians).

2.3. Unsupervised domain adaptation

Unsupervised domain adaptation (UDA) [27–29] aims to adapt a model trained on a labeled source domain, often synthetic data, to a target domain of real-world data. Among existing UDA methods, such as adversarial training [37–39] and self-training (ST) [30–32,40], ST methods have been recognized as a stable and effective technique for UDA, attaining state-of-the-art performance on image semantic segmentation tasks [30]. In ST, the network is trained using labeled source data and unlabeled target data with pseudo-labels that are predicted with confidence exceeding a predefined threshold. Zou et al. [32] proposed a ST-based UDA framework termed class-balanced self-training (CBST) for image semantic segmentation. To overcome the issue of class imbalance during domain transfer, CBST normalizes class-wise confidence scores to produce pseudo-labels with a balanced class distribution. Furthermore, an empirical and deterministic threshold updating rule based on quantiles, termed a self-paced learning policy, was introduced to automate the pseudo-labeling process.

Hoyer et al. [30] developed DAFormer, a framework for performing UDA with transformer-based image semantic segmentation architectures. Pseudo-labels are calculated online during training using a fixed threshold. To avoid adaptation instability and overfitting to the source domain, three training strategies for ST are integrated within the framework, including rare class sampling, thing-class ImageNet feature distance, and learning rate warmup. Pang et al. [33] applied the DAFormer framework to image semantic segmentation of high-speed railway viaducts and achieved performance close to that of supervised learning with real-world labeled data, facilitating bridge inspection tasks with limited manual efforts. These successful results are based on the high-quality source domain representation that enables reliable extraction of pseudo-labels with fixed or predetermined confidence thresholds.

UDA for 3D semantic and instance segmentation remains relatively limited [34]. Wu et al. [41] proposed a UDA pipeline for semantic segmentation of cars and pedestrians in road scenes using LiDAR point clouds. The approach leverages synthetic data during training and achieved improved performance on real-world datasets by employing three strategies to mitigate domain shift: learned intensity rendering, geodesic correlation alignment, and progressive domain calibration. UDA for 3D instance segmentation is even less explored in the literature. Rozenberszki et al. [42] employed a non-deep-learning 3D graph cut algorithm to generate pseudo-instance labels from 2D and 3D self-supervised features, enabling relatively accurate segmentation of major objects of interest, which could then be used for unsupervised learning. Consistent among these existing approaches is that the methods require reliable pseudo-labels derived from well-represented source domain data or relatively accurate candidate segmentations realized by other non-learning-based algorithms.

Direct applications of these UDA approaches to bridge point cloud semantic and instance segmentation are challenging because reliable pseudo-labels (or instance proposals) are hard to obtain under severe class imbalance and significant distribution shifts inherent in the problem. Bridge components appear in significantly different scales – for example, a deck typically appears as a single huge instance with high point density, while columns appear as thin and small instances. The point cloud quality of different parts of bridges is also influenced significantly by specific configurations of data collection. For example, careful LiDAR-based data collection can achieve relatively uniform point densities, while SfM-based data collection depends strongly on surface textures, visibility, and image capturing distances/overlap ratios, leading to highly non-uniform and noisy point distributions. Determining reasonable confidence thresholding rules in ST is difficult under severe class imbalance and distribution shift caused by these factors, combined with the underrepresentation of synthetic data discussed in Section 2.2.

2.4. Research gap and objective

Despite recent advances, bridge point cloud instance segmentation still faces significant challenges. Neither heuristic-based nor DL-based approaches can easily be scaled up to algorithms that can segment general real-world bridge point cloud data at the instance level, because of the strict adherence to the geometric assumptions and difficulty in collecting large-scale and diverse point clouds with ground truth instance annotations. Although synthetic data has been used to augment training sets, existing research imposes significant simplifications in terms of bridge types, point cloud quality, and labeling schemes. These simplifications lead to severe domain gaps between synthetic and real-world data, limiting the effectiveness of the synthetic data augmentation. Advances in UDA in the related fields (e.g., 2D semantic segmentation of driving scenes) can potentially be incorporated to compensate for this domain gap; however, its application to bridge point cloud instance segmentation remains challenging, because of severe class imbalance, domain shifts, and dataset scarcity (collecting and sharing relevant datasets are significantly more difficult than general problems, e.g., indoor scenes/objects). These challenges highlight the need for an enhanced UDA-based framework for bridge point cloud instance segmentation, encompassing synthetic data preparation, instance segmentation algorithms, and robust UDA algorithms.

This research proposes a UDA framework for deep learning (DL)-based instance segmentation of bridge point clouds with severe class imbalance and domain shift. A synthetic data generation method is first developed to produce instance-labeled point clouds across six bridge types. Additionally, a self-training (ST) approach, termed class-balanced dynamic thresholding (CBDT), is introduced to determine class-wise confidence thresholds for pseudo-labeling in a data-driven manner. By potentially eliminating time-consuming and costly ground truth instance-level annotations of real-world point clouds, this research facilitates efforts to scale up 3D instance segmentation algorithms to broader ranges of bridges, components, and data collection configurations (e.g., varying degrees of data quality).

3. Methodology

This research proposes a UDA approach for DL-based bridge point cloud instance segmentation under severe class imbalance and distribution shifts by combining the following three steps: synthetic point

cloud data generation with instance labels, UDA-based 3D instance segmentation, and class-balanced dynamic thresholding (CBDT) for pseudo-labeling during ST. The details of these steps are discussed in the following sections.

3.1. Synthetic point cloud data generation with instance labels

The overview of the approach for synthetic point cloud data generation for bridge component instance segmentation is shown in Fig. 1. The approach begins with the random and automated generation of bridges in a computer graphics (CG)-based synthetic environment [35]. Structure from Motion (SfM) is then applied to rendered synthetic images to generate a 3D point cloud dataset with semantic labels [26]. Finally, instance labels are automatically assigned based on point-to-mesh proximity. By integrating these three steps into a single framework, the approach enables fully automated synthetic data generation for bridge component instance segmentation.

3.1.1. Bridge geometry generation in a CG-based synthetic environment

This research employs the Random Bridge Generator (RBG) [35] to generate models of six types of railway bridges in a CG-based synthetic environment. RBG supports the random generation of six types of bridges by randomly assigning various types of bridge geometric parameters. Realistic 3D environments are created by assigning realistic textures to each component and placing synthetic bridges at random locations within the background models of real cities (Chicago in the USA, Kobe in Japan, and Lugano in Switzerland) imported from Google 3D Cities [43].

3.1.2. SfM-based 3D reconstruction with semantic labels

Based on the authors' preliminary work [26], this research simulates UAV-based image collection in synthetic bridge environments and reconstructs bridge point cloud data from the collected synthetic images. Random yet plausible camera paths are generated to collect global bridge rendered images [44], including the scenes, ground truth labels, and depth maps. The SfM method is employed to reconstruct 3D point cloud data from these images, with semantic labels obtained by mapping 2D labels to the 3D points. This pipeline effectively emulates real UAV surveys. Structural occlusions and inaccessible UAV viewpoints result in sparsity at component connections and non-uniform point distributions across identical components, as illustrated in Fig. 2. In addition, inevitable image matching errors and camera alignment inaccuracies introduce noise into the SfM outputs. To ensure the quality of the generated synthetic dataset, this research adopts the hyperparameters setting in [26]: (1) cases with more than 40 % of invalid camera alignment during SfM process are discarded, (2) point uncertainty is defined as the error between the *SfM-estimated* point depths and the *ground truth* depths obtained from rendered depth maps in the synthetic environment, and points in each point cloud are filtered using a uncertainty threshold of 0.1. This SfM-based dataset models certain real-world complexities, such as the non-uniform density of points and more severe occlusions of structural connections, compared with mesh-sampled and LiDAR-based datasets.

3.1.3. Instance label generation

This research automatically assigns instance labels to point clouds based on point-to-mesh proximity. The algorithm aligns semantic labels in the reconstructed point cloud with corresponding RBG-generated component meshes. For each semantic class, the algorithm calculates the distance from each point to all corresponding component meshes belonging to that class. Each point is assigned to the nearest component surface, and points associated with the same mesh are grouped into a single instance, as shown in Fig. 1.

3.2. UDA-based 3D instance segmentation approach

This research proposes an unsupervised domain adaptation (UDA)-based 3D instance segmentation approach tailored for two-stage instance segmentation models. To validate the proposed framework, SoftGroup++ [18,19], an advanced two-stage instance segmentation model, is adopted as the experimental framework. The following subsections present the overall framework, the adopted model, and the training strategy.

3.2.1. Overall UDA framework

The main challenge in applying UDA to instance segmentation of bridge point clouds lies in obtaining reliable pseudo-instance labels under severe class imbalance (different structural components vary widely in size and quality within the point clouds, depending strongly on bridge geometry and data acquisition settings) and domain shifts (synthetic data does not perfectly represent real-world data).

To mitigate these challenges, this research integrates UDA into deep learning (DL)-based point cloud instance segmentation through a two-stage framework (Fig. 3). The central idea is to integrate UDA into the

Fig. 1. Illustration of the synthetic bridge point clouds generation process.

(a) Noisy SfM outputs (due to image matching errors and camera misalignments)

(b) Sparsity at structural connections (e.g., sparsity at deck-pier joints, caused by occlusions)

(c) Non-uniform density across identical components (e.g., sparsity in some piers, caused by limited UAV viewpoints)

Fig. 2. Characteristics of the synthetic bridge point clouds generated by SfM.

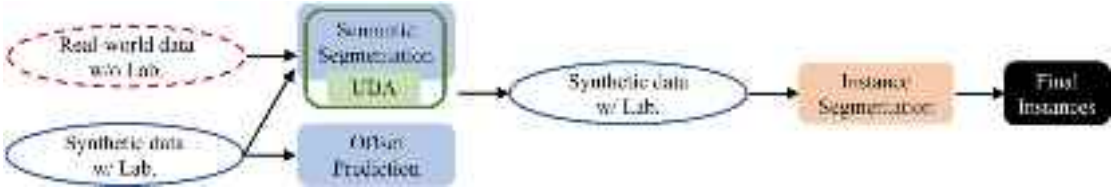


Fig. 3. Architecture of a two-stage UDA-based 3D instance segmentation framework.

semantic segmentation stage, which commonly serves as an intermediate step for feature extraction and preliminary grouping. By improving the accuracy of the semantic segmentation module using unlabeled real-world data and leveraging relatively high-quality semantic pseudo-labels, the proposed framework enhances instance segmentation performance while circumventing the need for direct pseudo-instance labels.

3.2.2. Instance segmentation model

To implement and verify the proposed framework, SoftGroup++ [18,19], a two-stage instance segmentation algorithm, is adopted. SoftGroup++ integrates common proposal-based and grouping-based methods, offering strong performance and high computational efficiency on general indoor scene point cloud datasets [45–47]. As illustrated in Fig. 4, SoftGroup++ performs instance segmentation in three key stages: (1) point-wise prediction, where a 3D U-Net [48,49] extracts multi-scale features from voxelized point clouds, followed by two multi-layer perceptron (MLP) branches that predict point-wise semantic scores and offset vectors (a vector that brings each point to the associated instance centers), (2) preliminary instance proposal generation by a technique termed soft grouping, where each point is associated with multiple classes based on soft semantic scores (rather than hard one-hot predictions), followed by the shifting using the predicted offset vector and the class-wise clustering, (3) instance refinement, where another smaller U-Net predicts instance masks, mask scores, and classification scores for each proposal. Combining these results, the network can generate and refine instances in an end-to-end manner, while having clearly defined intermediate steps, such as semantic segmentation and offset vector prediction.

3.2.3. Training strategy

Based on this point cloud instance segmentation algorithm with a clearly defined semantic segmentation step, this research proposes to first train the semantic segmentation part ("bottom-up grouping" part indicated in Fig. 4) in a UDA manner. Two datasets are leveraged in this stage: large-scale synthetic data with *ground truth* semantic annotations and offset vectors (source domain), and *unannotated* real-world data (target domain). During training, the loss for the semantic branch is computed by combining the standard cross-entropy loss obtained from the source domain data with ground truth semantic annotations, L_S^{semantic}, and self-training loss obtained from the unannotated target domain data, L_T^{semantic} (details are discussed in Section 3.3.1). At the same time, the ℓ_1 regression loss for the offset branch is computed using the source domain data only. The 3D U-net with separate heads for semantic score and offset vector predictions is trained to minimize this combined loss function. Once this stage is completed, the backbone weights are frozen.

The proposed approach then trains the top-down instance refinement

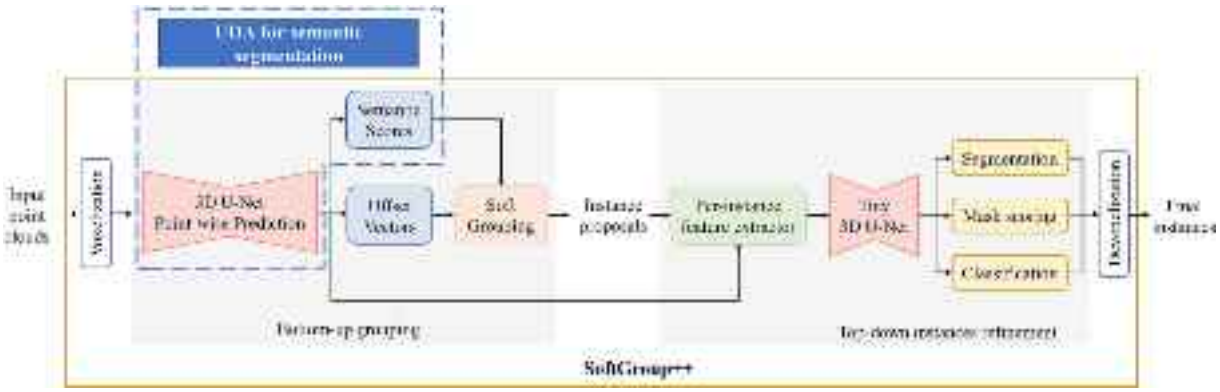


Fig. 4. Architecture of the proposed framework based on SoftGroup++.

networks using the synthetic dataset with semantic and instance annotations. The segmentation, mask scoring, and classification branches are optimized using a combination of binary cross-entropy, ℓ_2 regression loss, and cross-entropy. The training in this stage can benefit from UDA-enhanced semantic representations, while avoiding the need for pseudo-instance labels, which are often unreliable and difficult to obtain for the bridge component instance segmentation problem.

3.3. Self-training with class-balanced dynamic thresholding

The proposed UDA method is based on the self-training (ST) framework, and implemented in the three steps: (1) extending the high-performance ST framework for image semantic segmentation tasks, termed DAFormer [30], to 3D point cloud semantic segmentation tasks, (2) incorporating a mechanism to compensate for the severe class imbalance observed in the bridge component instance segmentation problem, with the help of class-balanced self-training (CBST) method [32], and (3) addressing the limitation of the existing pseudo-labeling process [32] by dynamically adjusting the class-wise pseudo-labeling thresholds in a data-driven manner - the method termed class-balanced dynamic thresholding (CBDT) herein. This combination is designed to support UDA under severe class imbalance and distribution shifts in bridge point cloud data.

3.3.1. Self-training framework

The UDA framework in this research is inspired by the DAFormer framework [30], which was originally proposed for image semantic segmentation. In [30], the image semantic segmentation model g_θ with network weights θ, which takes an input image $\boldsymbol{x} \in \mathbb{R}^{H\times W\times 3}$ and predicts semantic scores $\boldsymbol{s} \in \mathbb{R}^{H\times W\times N_{Class}}$, where H and W are the image height and width. The source domain data consists of a set of images $\boldsymbol{X}_S = \left\{\boldsymbol{x}_S^{(i)}\right\}_{i=1}^{N_S} \subset \mathbb{R}^{H\times W}$ and the associated ground truth one-hot labels $\boldsymbol{S}_S^* = \left\{\boldsymbol{s}_S^{*(i)}\right\}_{i=1}^{N_S} \subset \mathbb{R}^{H\times W\times N_{Class}}$, where N_s is the number of images in the source domain. Similarly, the target domain data is denoted as $\boldsymbol{X}_T = \left\{\boldsymbol{x}_T^{(i)}\right\}_{i=1}^{N_T} \subset \mathbb{R}^{H\times W\times 3}$, where N_T is the number of images in the target domain.

This research extends the DAFormer framework by reinterpreting g_θ as a point cloud semantic segmentation algorithm (SoftGroup++) that consumes point cloud data $\boldsymbol{x} \in \mathbb{R}^{N\times 3}$ and predicts the per-point semantic scores $\boldsymbol{s} = g_\theta(\boldsymbol{x}) \in \mathbb{R}^{N\times N_{Class}}$, where N is the total number of points in the batch. For source domain point cloud data, the cross-entropy (CE) loss for the supervised learning is defined as:

$$L_S^{\text{semantic}} = \frac{1}{N_S}\sum_{i=1}^{N_S} CE\left(\boldsymbol{s}_S^{(i)}, \boldsymbol{s}_S^{*(i)}\right), \tag{1}$$

where N_S is the total number of source domain points in the batch.

Following DAFormer, the ST approach employs a teacher-student architecture [50] to produce pseudo-labels for the target domain data. The teacher network $h_\varnothing$ shares the same architecture as the student network g_θ. During training, the teacher network's weights $\varnothing$ are updated by the exponential moving average (EMA) of the student network's weights θ:

$$\varnothing_{t+1} \leftarrow \alpha\varnothing_t + (1-\alpha)\theta_t. \tag{2}$$

where $\varnothing_t$ and θ_t are the weights of teacher and student networks at iteration t, and α is a smoothing coefficient empirically set to 0.99 [50]. Pseudo-labels for the target domain are generated using the teacher network $h_\varnothing$:

$$p_T^{(i,c)} = \left[c = \underset{c'}{\text{argmax}}\, h_\varnothing\left(\boldsymbol{x}_T^i\right)^{(c')}\right], \tag{3}$$

where $[\cdot]$ denotes the Iverson bracket. Additionally, a confidence estimate for pseudo-labels is produced using the proportion of points or pixels whose maximum softmax scores exceed a threshold $\tau = 0.968$ [31]:

$$q_T = \frac{1}{N_T}\sum_{i=1}^{N_T}\left[\underset{c'}{max} h_\varnothing\left(\boldsymbol{x}_T^{(i)}\right)^{(c')} > \tau\right], \tag{4}$$

where N_T is the total number of target domain points in the batch. The pseudo-labels and their quality estimates are used to train the network g_θ on the target domain with the self-training loss:

$$L_T^{\text{semantic}} = -\frac{1}{N_T}\sum_{i=1}^{N_T}\sum_{c=1}^{N_{Class}} q_T\, p_T^{(i,c)} \log s_T^{(i)}. \tag{5}$$

The overall UDA loss $L^{semantic}$ is the sum of these loss components

$$L^{\text{semantic}} = L_S^{\text{semantic}} + L_T^{\text{semantic}}. \tag{6}$$

Preliminary experiments with the extended DAFormer showed that the UDA performance is sensitive to the fixed threshold τ applied to the softmax scores of the teacher network. Although the value $\tau = 0.968$ used in [30] was appropriate for the UDA problem investigated therein, this research found that the same threshold led to highly suboptimal results in the bridge component instance segmentation problem. This could be explained by the severe class imbalance in the dataset, where the fixed threshold disproportionately favored frequent classes while neglecting rare ones.

This research partially addresses this issue by incorporating class-balanced self-training (CBST) [32] into the UDA process. CBST defines per-class thresholds τ_c, and normalizes confidence scores at the class level to compensate for this imbalance issue. In this method, the threshold for class c is defined as τ_c. The threshold τ_c is used to scale the teacher network's softmax scores, yielding the adjusted probability $\widehat{h}_\varnothing\left(\boldsymbol{x}_T^{(i)}\right)^{(c)}$:

$$\widehat{h}_\varnothing\left(\boldsymbol{x}_T^{(i)}\right)^{(c)} = h_\varnothing\left(\boldsymbol{x}_T^{(i)}\right)^{(c)} \Big/ \tau_c. \tag{7}$$

Based on these adjusted probabilities, the pseudo-labels are produced as follows:

$$\widehat{p}_T^{(i,c)} = \left[c = \underset{c'}{\text{argmax}}\, \widehat{h}_\varnothing\left(\boldsymbol{x}_T^i\right)^{(c')}\right] \cdot \left[\widehat{h}_\varnothing\left(\boldsymbol{x}_T^i\right)^{(c)} > 1\right]. \tag{8}$$

This approach balances the pseudo-label assignment toward classes that have relatively low scores but high within-class confidence. The pseudo-label of any point is filtered out only when the adjusted probabilities for all classes are smaller than 1. When multiple classes satisfy the condition $\widehat{h}_\varnothing\left(\boldsymbol{x}_T^{(i)}\right)^{(c')} > 1$, the class with the maximum adjusted probability is selected as the final pseudo-label. Accordingly, confidence estimate $\widehat{q}_T$ computed by

$$\widehat{q}_T = \frac{1}{N_T}\sum_{i=1}^{N_T}\left[\max_{c'}\left(\widehat{h}_\varnothing\left(\boldsymbol{x}_T^{(i)}\right)^{(c')}\right) > 1\right]. \tag{9}$$

In CBST, per-class threshold τ_c $(c = 1, 2, \ldots, N_{Class})$ is determined using self-paced learning policy [32], where τ_c corresponds to the top $p \times 100\%$ adjusted probabilities of predicted points for class c. The proportion variable p is initially set to 0.2 and is increased by 0.005 in each epoch with a maximum limit of 0.5. While this approach accounts for the different probability distributions of various classes, it relies on an empirical and deterministic threshold updating rule based on quantiles, without considering the current state of the data or network. The method assumes that the predicted points for each class contain at least $p \times$ 100% true positives (TPs), where the variable p is hard-coded. However, this assumption may not hold consistently across classes. As

illustrated in Fig. 5, for rare classes with low recognition accuracy, the top 100p% quantile of the predicted softmax probability may have little overlap with TPs, and the resulting threshold may be highly unreliable. In the extreme case where no TPs exist in a given data batch (which is possible in this research), the self-paced learning policy looks up false positives (FPs) to determine the threshold (if any FPs are present) or fails when no predictions exist. In contrast, for frequent and well-represented classes, the top 100p% may occupy a small fraction of TPs, lowering learning effectiveness. Consequently, for rare classes, this policy tends to include more FPs, while for frequent and well-represented classes, it may filter out a substantial portion of TPs. This identical updating rule across classes can lead to suboptimal selection of training samples and ultimately hinder the overall training performance.

3.3.2. Class-balanced dynamic thresholding

To overcome the limitations of the self-paced learning policy in CBST, this research proposes a class-balanced dynamic thresholding (CBDT) method. CBDT adaptively updates the pseudo-label threshold at each iteration by leveraging the softmax probabilities of all points in the point cloud, including points that belong to a certain class and points that do not. The ground truth labels (whether the points belong to that class or not) are not available because the data comes from an unannotated target domain. The CBDT method should infer (1) which points in the point cloud come from the current class considered, and (2) which value of softmax probability would be appropriate to extract reliable pseudo-labels for that class. To achieve this, the proposed approach applies Gaussian Mixture Model (GMM) fitting using the expectation-maximization (EM) algorithm. The estimation of latent variables in GMM provides a natural solution to point (1), and identifying the peak location of the major Gaussian component on the positive side of the inverse sigmoid mapped softmax probability (termed the "positive dominant peak" in this research) can effectively serve as the threshold determination process discussed in point (2). The detailed description of the approach is as follows.

As shown in Fig. 6(a), the histogram of the softmax probabilities significantly deviates from the Gaussian bell-shaped curve, with particularly poor fits near the bounds (Fig. 6(b)), making it inappropriate to use GMM components to represent the data. To improve the fit, the softmax probabilities $\boldsymbol{s}^{(c)} = \left\{ s_S^{(i,c)} \right\}_{i=1}^{N} \subset [0,1]$ for each class c assigned by the current teacher network are mapped into the range $(-\infty, \infty)$ by the inverse sigmoid function

$$\widehat{s}^{(i,c)} = \sigma^{-1}\left(s^{(i,c)}\right) = -\ln\left(1/s^{(i,c)} - 1\right), \tag{10}$$

where $s^{(i,c)}$ and $\widehat{s}^{(i,c)}$ denote the original and mapped probabilities for point i and class c. Note that $\boldsymbol{s}^{(c)}$ includes probabilities for all points, not only the points predicted as class c. As shown in Fig. 6, the distribution of mapped probabilities $\widehat{\boldsymbol{s}}^{(c)}$ aligns more closely with the GMM, which can be fitted by the expectation maximization algorithm.

Each Gaussian component in the fitted GMM can be interpreted as a cluster of points with similar confidence levels of belonging to class c. TP predictions tend to form a Gaussian component with a mean greater than zero (the original probability before mapping is larger than 0.5), while the predictions not belonging to class c are likely associated with Gaussian components with means less than zero. The CBDT method selects the dominant positive component, which is the Gaussian component with the highest peak among the components with positive mean values. The mean value of the dominant positive component, μ_c^t, is then mapped back to the original probability space by the sigmoid function to obtain the class-specific intermediate threshold $\overline{\tau}_c^t$ at iteration t:

$$\overline{\tau}_c^t = {}^{1}\Big/\left(1 + e^{-\mu_c^t}\right). \tag{11}$$

To avoid noisy GMM fitting results from negatively affecting the ST process, the CBDT method assesses the quality of the dominant positive component by calculating the Wasserstein distance (w_{dist}) between the fitted dominant positive component P and the part of the histogram Q associated with that component:

$$w_{\text{dist}}(P,Q) = \inf_{\gamma \in \Gamma(P,Q)} \int_{R^2} |x-y| \, d\gamma(x,y), \tag{12}$$

where $\Gamma(P,Q)$ is the set of all joint distributions $\gamma(x,y)$ with marginals P and Q. Fig. 7 shows examples of GMM with eight components and the w_{dist} evaluated for the dominant positive component. A small w_{dist} (e.g., $w_{\text{dist}} = 0.081$) indicates a high-quality fit with a clear and distinct Gaussian-like peak on the positive side, which can be interpreted as a cluster of points that are highly likely to belong to the selected class. A larger w_{dist} (e.g., $w_{\text{dist}} = 0.299$) suggest the peak is distinct but not well-represented as a Gaussian. An even larger w_{dist} (e.g., $w_{\text{dist}} = 0.488$) indicates a relatively poor fit, where the dominant component fails to represent a meaningful peak, allowing the algorithm to reject such unreliable peaks in a data-driven manner.

Based on the results of preliminary experiments, $\overline{\tau}_c^t$ is accepted for updating the class-specific threshold τ_c^t at iteration t with EMA only when the w_{dist} is less than 0.3:

$$\tau_c^t \leftarrow \alpha \tau_c^{t-1} + (1-\alpha)\overline{\tau}_c^t, \tag{13}$$

where α is the smoothing coefficient, empirically set to 0.99. Otherwise, it remains unchanged: $\tau_c^t \leftarrow \tau_c^{t-1}$.

Algorithm 1 summarizes the proposed CBDT method, which adaptively adjusts class-wise thresholds based on data distribution, enabling effective pseudo-label generation for class-balanced self-training. At every iteration during training, the thresholds are updated by Algorithm 1, and then used to compute the adjusted probability in CBST by Eq. (7). The pseudo-labels are filtered for training based on Eq. (8), with their

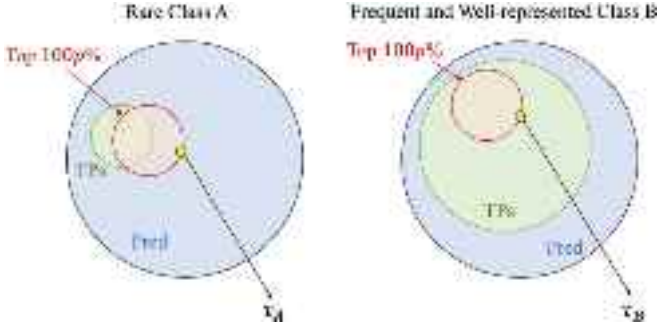
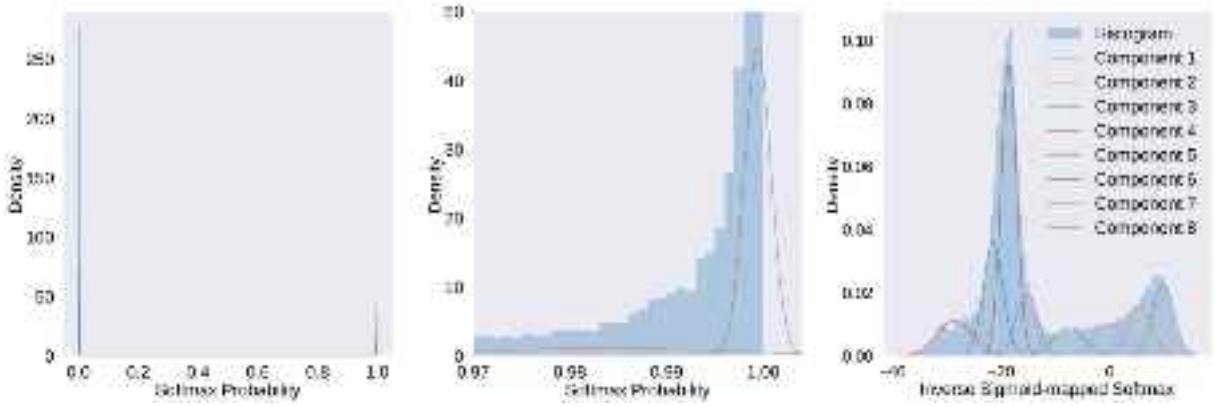


Fig. 5. Illustration of the relationship between points above the threshold (updated by the self-paced learning policy) and true positives in rare and confident classes.



(a) Histogram of softmax probability (b) Histogram of softmax probability (plotted for 0.97<x<1.01) (c) Histogram of mapped probability

Fig. 6. Original and mapped probability distributions with GMM fits.

confidence estimated by Eq. (9), from which the self-training loss is computed in Eq. (3).

Algorithm 1. Determination of τ_c in CBDT.

Input : Teacher network $\boldsymbol{h}_{\emptyset}$, target point clouds $\{\boldsymbol{x}_T^{(i)}\}_{i=1}^{N}$, initial thresholds $\{\tau_c^0\}_{c=1}^{C}$, smoothing coefficient α

Output : Updated thresholds $\{\tau_c^t\}_{c=1}^{C}$

for $c = 1$ to C **do**
  $\hat{\boldsymbol{s}}^{(c)} = []$
  for $i = 1$ to N **do**
    $\boldsymbol{s}^{(i)} = h(\emptyset, \boldsymbol{x}_T^{(i)})$
    $s^{(i,c)} = \boldsymbol{s}^{(i)}[c]$
    $\hat{s}^{(i,c)} = \text{inverse\_sigmoid}(s^{(i,c)})$
    $\hat{\boldsymbol{s}}^{(c)}.\text{append}(\hat{s}^{(i,c)})$
  end
  GMM = fit_gaussian_mixture($\hat{\boldsymbol{s}}^{(c)}$, n_components = 8)
  dominant_comp = get_highest_peak_with_positive_mean(GMM)
  μ_c^t = dominant_comp.mean
  $\overline{\tau}_c^t = \text{sigmoid}(\mu_c^t)$
  w_{dist} = wasserstein_distance(dominant_comp, $\hat{\boldsymbol{s}}^{(c)}$)
  if $w_{\text{dist}} < 0.3$:
    $\tau_c^t \leftarrow \alpha\tau_c^{t-1} + (1-\alpha)\overline{\tau}_c^t$
  else:
    $\tau_c^t \leftarrow \tau_c^{t-1}$
end
return $\{\tau_c^t\}_{c=1}^{C}$

4. Experimental validation

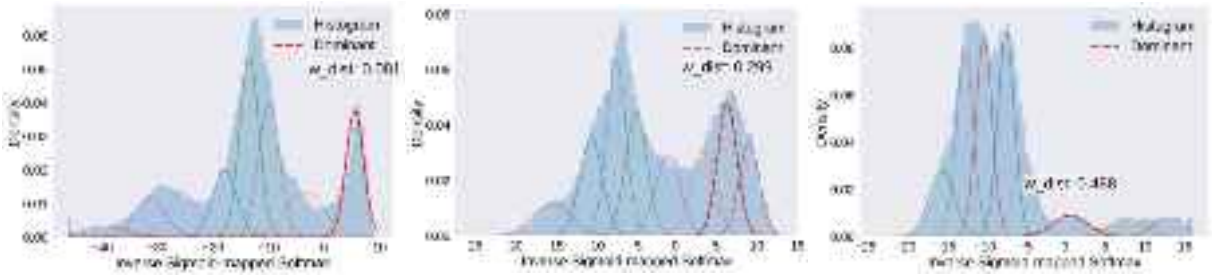
This section presents the experimental validation of the proposed method, including the datasets and experimental settings.

4.1. Datasets

The proposed method is evaluated using bridge datasets across one synthetic and four real-world domains. The synthetic railway bridge dataset, S1, is generated by the approach discussed in Section 3.1. The real-world datasets include R1 [51,52] (7 RC highway bridges in Ulsan, South Korea, collected by LiDAR), R2 [53,54] (4 RC railway bridges across four cities, Portugal, collected by a hybrid method based on LiDAR and SfM), R3 [8] (10 RC highway bridges around Cambridgeshire, United Kingdom (UK), collected by LiDAR) and R4 (2 RC highway bridges in Shanghai, China, collected by LiDAR). These datasets are characterized by different data acquisition methods, bridge types, structural dimensions, and geographic locations, providing support for evaluating the performance of the proposed approach in both domain adaptation (testing data is from the target domain considered during training) and domain generalization (testing domain is different from the target domain considered during training) scenarios.

4.1.1. Synthetic bridge data (S1)

This research employs the approach discussed in Section 3.1 to randomly generate synthetic point cloud data for 199 railway bridges with six types: slab, beam, girder, arch, cable-stayed, and suspension



(a) High-quality fit $w_{\text{dist}} = 0.081$ (b) Acceptable fit $w_{\text{dist}} = 0.299$ (c) Rejected fit $w_{\text{dist}} = 0.488$

Fig. 7. Association of GMM fit quality in eight-component mixtures with w_{dist}.

bridges. The dataset consists of global sparse point cloud data obtained through SfM reconstruction with semantic and instance labels. This research merges decks, bearings, girders, beams, tracks, sleepers, and slabs into a unified deck class. The final synthetic dataset includes background and eight component classes, with instance labels aligned to the merged semantic classes. The size ranges of the synthetic bridge point cloud data are listed in Table 1, and examples of the processed synthetic bridge point clouds are shown in Fig. 8.

4.1.2. Real-world bridge data (R1-R4)

The proposed framework for bridge component recognition was evaluated on four bridge domains that differ in geographical location, data acquisition methods, bridge types, and structural dimensions and shapes, as summarized in Table 2 and illustrated in Fig. 9. A total of 23 bridges from four datasets (R1–R4) were utilized, with point clouds capturing both structural components and surrounding background, acquired using various laser scanning and photogrammetry techniques.

R1 [51,52] **– Ulsan, South Korea**: 7 highway slab and beam bridges were scanned using a RIEGL VZ-1000 3D terrestrial laser scanner and a digital camera. The bridges range in total length from 54 to 272 m, spans from 17 to 38 m, and heights from 7 to 39 m, featuring multiple piers of varying heights and curved decks.

R2 [53,54] **– Grândola, Rio Tinto, Mouquim, Aveiro, Portugal**: 4 railway beam bridges were scanned using terrestrial/mobile laser scanners and UAV photogrammetry, integrating both UAV and terrestrial scanning to generate precise, high-quality 3D models. The bridges range in total length from 30 to 100 m, spans from 10 to 27 m, and heights from 3 to 11.75 m, with piers of varying heights.

R3 [8] **– Cambridgeshire, United Kingdom**: 10 highway slab and beam bridges were scanned using a FARO Focus 3D X330 terrestrial laser scanner. The bridges range in total length from 55 to 91 m, spans from 13 to 35 m, and heights from 5 to 8 m, with most bridges featuring multiple piers of varying heights and horizontally curved decks.

R4 – Shanghai, China: 2 highway slab and beam bridges were scanned using UAV LiDAR, with total lengths from 35 to 200 m, spans from 5 to 25 m, and heights from 8 to 30 m. The two bridges differ considerably; one is a pedestrian overpass, while the other is a tall cross-river bridge.

These differences of **R1**-**R4** lead to highly distinct point cloud characteristics across the domains, including variations in point density, component categories and positions, and structural features.

This research pre-processes the raw data by (1) extracting the main structure of the bridge without irrelevant surroundings, (2) downsampling the point cloud to 100,000 points, and (3) assigning a uniform white color to the points if the raw data lacks RGB color information (applied to R1 and R4).

The pre-processed bridge point cloud data is then segmented into 30-m sections along the longitudinal axis, aligning with the 30- to 40-m-long slab and beam bridges in the synthetic dataset. Segments do not overlap longitudinally, except for the final segment, which extends 30 m backward from the end of the bridge. After segmenting the data, segments with at least 1000 points are kept and used in the subsequent analyses, resulting in 28, 10, 21, and 7 segments for R1, R2, R3, and R4, respectively. The bridge point cloud data is annotated into semantic and instance classes to enable validations of the proposed UDA approach. The annotation process uses CloudCompare [56], an open-source software designed for 3D point clouds and triangular meshes processing. This research defines five semantic classes that are consistent with class definitions in the synthetic dataset: background, pier, railing, deck, and piercap. Fig. 9 illustrates examples of segmented real-world bridge point cloud data, along with ground truth semantic and instance labels.

Table 1
Bridge parameters.

Bridge Type [55]	Total Length [m]	Span Length [m]	Height [m]
Slab/Beam/Girder bridge	[30, 40]	[8, 12]	[8, 10]
Arch bridge	[50, 55]	[8, 12]	[25, 30]
Suspension bridge	[180, 320]	[10, 15]	[7.5, 10]
Cable-stayed bridge	[45, 320]	[10, 15]	[8, 10]

4.1.3. Class distributions

Fig. 10 shows the component class distributions across datasets. In the synthetic dataset S1, piers and decks dominate, while other classes occupy only a marginal proportion, demonstrating severe class imbalance. In real-world datasets R1-R4, the background class proportion is significantly larger than that in S1, and other classes exhibit varying distributions across domains, causing substantial differences between real-world and synthetic domains.

The Imbalanced class distribution of the source domain and the distribution difference between source and target domains lead to varying transfer difficulties across classes, resulting in different prediction confidence levels in the target domain. Since ST selects pseudo-labels with high confidence, it tends to favor easy-to-transfer classes while neglecting others, leading to inferior adaptation performance in bridge component instance segmentation investigated in this research.

4.2. Experimental settings

Before training, all data are pre-processed by (1) applying rotations to align the bridge's longitudinal direction with the x-axis, the transversal direction with the y-axis, and the vertical direction with the z-axis, and (2) applying translation to set the origin at the mean x and y coordinates of all points in the point cloud, and the mean z coordinate of the background points. The datasets are randomly divided into 80 % (Split1) and 20 % (Split2). An overview of the dataset splits is shown in Table 3, with further details provided in the following two sections.

The semantic segmentation results are evaluated by mean Intersection over Union (mIoU) and accuracy (Acc) [57]. The results of offset vector prediction are evaluated by Mean Absolute Error (MAE). Instance segmentation results are evaluated by the standard Average Precision (AP) [58]. Here, AP_{50} and AP_{25} denote the scores with Intersection over

Fig. 8. Examples of the processed synthetic point clouds for six bridge types (first column), their corresponding ground truth semantic labels (second column), and ground truth instance labels (third column). The color bar at the bottom indicates the semantic class mapping, while distinct colors represent individual instances (black indicates background, which is excluded from instance counting)

Table 2
Summary of the real-world bridge data.

Dataset	No. of Bridges	Location	Data Acquisition	Bridge Type [55]
R1 [51,52]	7	Ulsan (South Korea)	RIEGL VZ-1000 3D terrestrial laser scanner & digital camera	Highway slab & beam bridge
R2 [53,54]	4	Grândola, Rio Tinto, Mouquim, Aveiro (Portugal)	Terrestrial/mobile laser scanners & UAV photogrammetry	Railway beam bridge
R3 [8]	10	Cambridgeshire (UK)	FARO Focus 3D X330 terrestrial laser scanner	Highway slab & beam bridge
R4	2	Shanghai (China)	UAV LiDAR	Highway slab & beam bridge

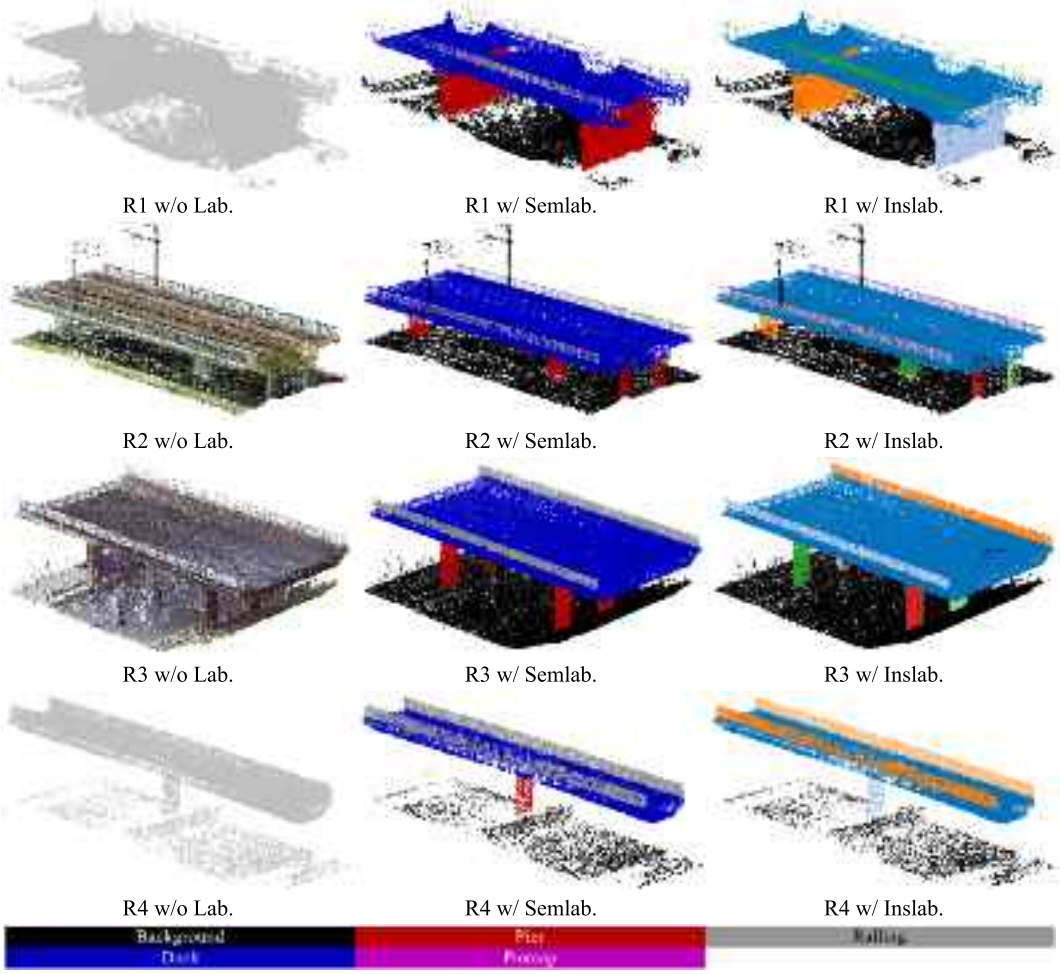


Fig. 9. Examples of segmented real-world bridge point clouds (first column), their corresponding ground truth semantic labels (second column), and ground truth instance labels (third column). The point cloud data without color information is displayed in uniform gray for visualization. The color bar at the bottom indicates the semantic class mapping, while distinct colors represent individual instances (black indicates background, which is excluded from instance counting).

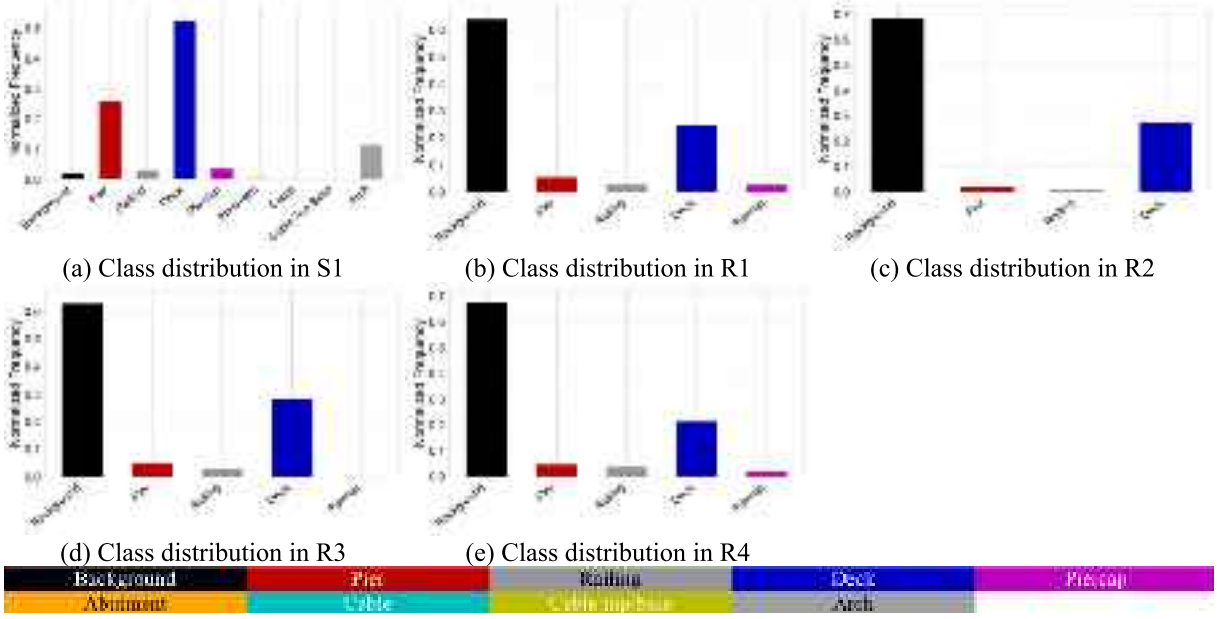


Fig. 10. Normalized distribution of component classes for each preprocessed dataset (color mapping shown in bar below).

Table 3
Dataset split overview and point cloud counts.

Dataset	Split1 (80 %)	Split2 (20 %)	All (100 %)
S1: Synthetic data	160	39	199
R1: Real-world data in Ulsan, South Korea [51,52]	22	6	28
R2: Real-world data across four cities, Portugal [53,54]	8	2	10
R3: Real-world data around Cambridgeshire, UK [8]	16	5	21
R4: Real-world data in Shanghai, China	5	2	7

Union (IoU) thresholds of 50 % and 25 %, respectively. Similarly, AP denotes the average scores with IoU thresholds ranging from 50 % to 95 % (with a step size of 5 %). The model runs on a PC equipped with NVIDIA GeForce RTX 4090 and RTX 3090 GPUs, each with 24 GB of memory.

The hyperparameters of the SoftGroup++ algorithm are carefully selected by training and validation on the synthetic dataset S1. Key hyperparameters include the scaling factor, grouping radius, and the score threshold. The scaling factor, which controls the balance between the segmentation resolution and computational cost by down-sampling point cloud data into voxels, was tested in the range of 5 to 20. The scaling factor of 16 yielded the highest mIoU for semantic segmentation. The grouping radius, determining the search range of the k-Nearest Neighbor (kNN) algorithm during the soft grouping stage, was adjusted between 0.4 and 1. A grouping radius of 0.6 resulted in the best instance segmentation across AP, $AP_{50,}$ and AP_{25} metrics. The score threshold, which affects the number of semantic classes each point is associated with during the soft grouping stage, was tuned from 0.1 to 0.6, with 0.4 achieving optimal performance. All subsequent experiments were conducted with these optimized hyperparameters.

The SoftGroup++ model is trained by Adam optimizer [59] for 128 epochs with a batch size of 4 for both source and target domain data. The learning rate is initialized to 0.002 and scheduled by a cosine annealing. Following DAFormer [30], a linear learning rate warmup is applied in the UDA stage, scaling the original learning rate at iteration t by t/t_{warm} during the warmup phase ($t \leq t_{warm} = 1500$). This research does not implement rare-class sampling or thing-class ImageNet feature distance proposed in the DAFormer, considering the significant difference of the problem investigated in this research from the UDA for image semantic segmentation in urban driving scenes [60]. Data augmentation includes random scaling (±40 % per axis), rotation around the z-axis (0 to 360 degrees), translation (±1 m per axis), and horizontal flipping (50 % probability).

For the ST using CBDT, the confidence thresholds τ_c for background, pier, railing, deck, piercap, and abutment are initialized to 0.968 and updated by the method introduced in Section 3.3.2. Since the real-world data in this research does not include cable, cable top/base, and arch classes, ST is not applied to those classes. A GMM with eight components is used to fit class-wise distributions of probabilities. The fit is considered acceptable if w_{dist} between the dominant positive component and its corresponding histogram part is less than 0.3. Before GMM fitting, points in each batch are down-sampled by a factor of 40 to improve computational efficiency.

This research validates the proposed approach by comparing networks trained in three configurations (Source-only, Mix, and Mix-UDA), as illustrated in Fig. 11. In the Source-Only configuration, standard supervised learning is performed using the annotated S1_Split1 dataset, serving as a baseline for evaluating the effectiveness of the proposed UDA framework. In the Mix configuration, the annotated R1_Split1 dataset is combined with S1_Split1 to train the semantic segmentation and offset prediction modules, while the top-down refinement network is trained exclusively on S1_Split1. This configuration represents the upper performance bound, as it leverages ground truth annotations from the target domain to generate preliminary instance proposals based on semantic scores and offset vectors obtained through supervised learning. In the Mix-UDA configuration, UDA is applied during semantic segmentation training, using labeled S1_Split1 and unlabeled R1_Split1, while the offset prediction and top-down refinement network are trained using S1_Split1.

5. Results and discussion

This section presents the results of the methodology described in Section 3, with qualitative analyses of the pseudo-labeling process and quantitative evaluations of domain adaptation and generalization performance.

5.1. Qualitative evaluation of pseudo-labeling in CBDT

This section qualitatively evaluates the performance of CBDT by analyzing the pseudo-labeling process for the target domain during training. Fig. 12 shows histograms decomposed into distributions of positive points (those belonging to the given class, i.e., $TP + FN$) and negative points (not belonging to the class, i.e., $TN + FP$) with the mean values of the dominant positive components obtained through GMM

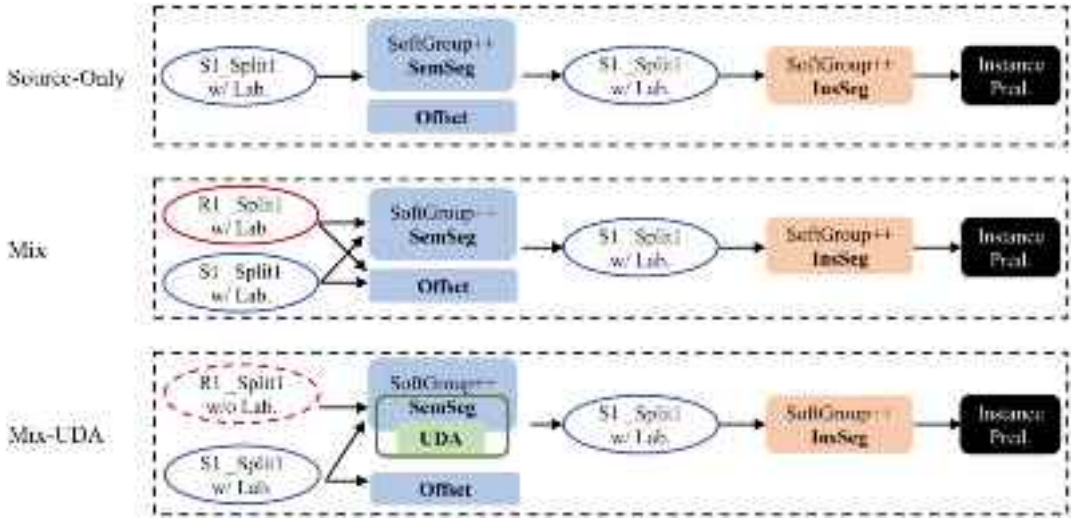


Fig. 11. Training configurations.

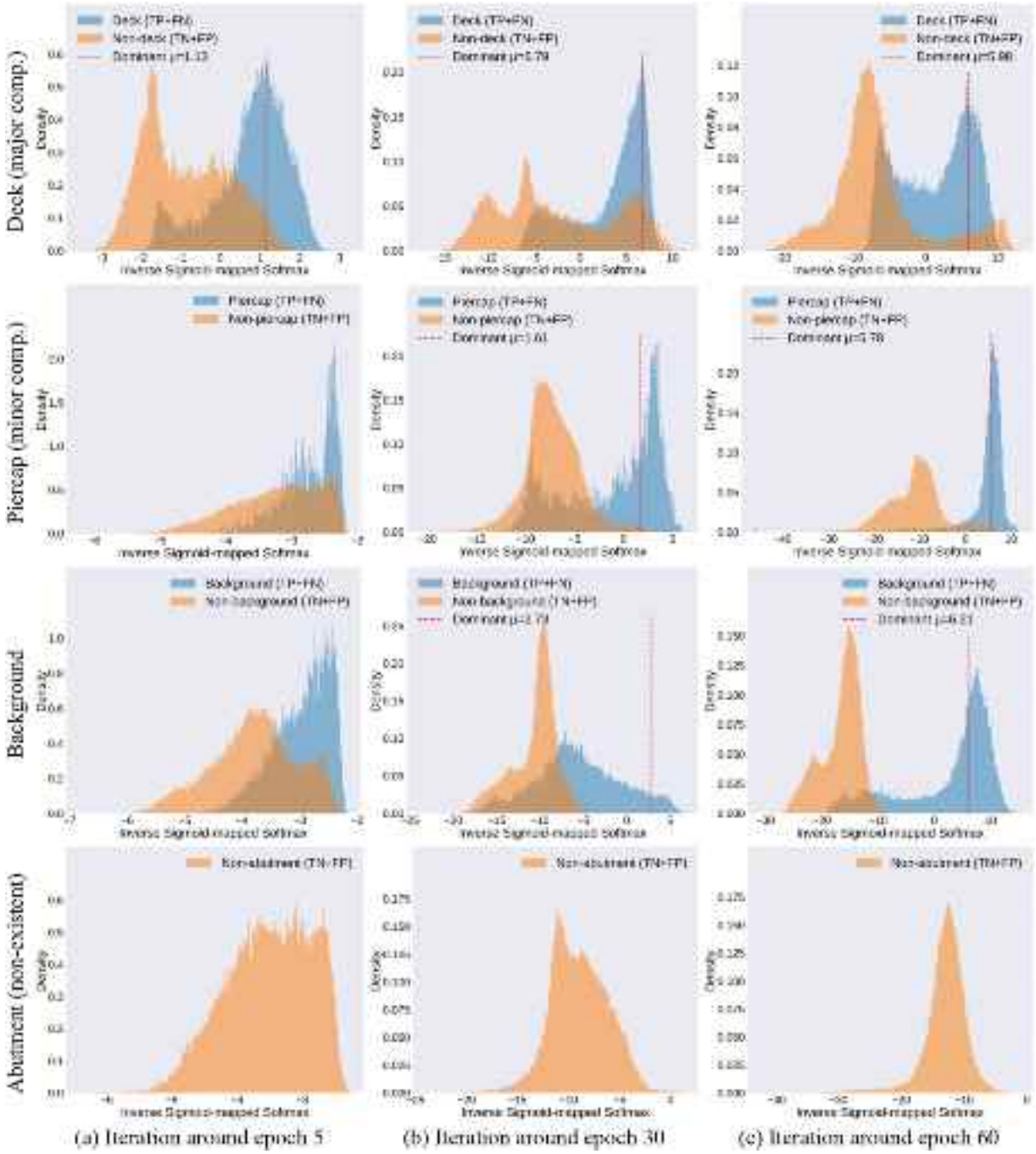


Fig. 12. Evolution of inverse sigmoid-mapped probabilities with GMM fitting (at iteration around epochs 5/30/60).

fitting (note that no annotation was used to determine the dominant positive components). These mean values are used to update class-specific thresholds, as explained in Section 3.3.2.

As training progresses, the separation between positive and negative distributions becomes increasingly clear, and a dominant positive component emerges. Once this component appears, most true positives (TP) are found well to the right of its mean value. This observation supports the effectiveness of using the mean of the dominant positive component to adjust class-specific thresholds.

For major structural components such as pier and deck, positive and negative distributions are well separated in early training stages, allowing the threshold to be meaningfully adjusted at the beginning. For minor components such as piercap and railing, as well as non-structural background, the overlap between positive and negative points is more significant at early stages, but gradually decreases as training continues. For the abutment class, which does not exist in the target domain, the GMM fitting typically produces no positive component (i.e., no points have pre-mapped probabilities greater than 0.5). As a result, the threshold for this class remains unchanged, which is consistent with the intended design.

Fig. 13 tracks the threshold evolution for each class. Consistent with the histogram trends in Fig. 12, thresholds are first updated for the major structural components (pier and deck classes), followed by minor components (piercap and railing) and non-structural background, with the non-existent abutment class adjusting last. The threshold update magnitude correlates with the peak location of the dominant positive component, and threshold values converge gradually as training progresses.

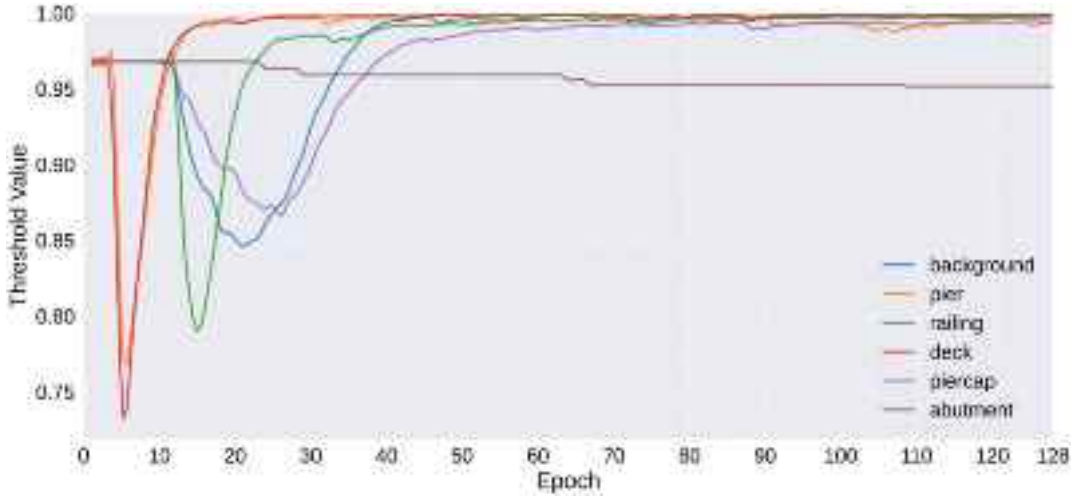
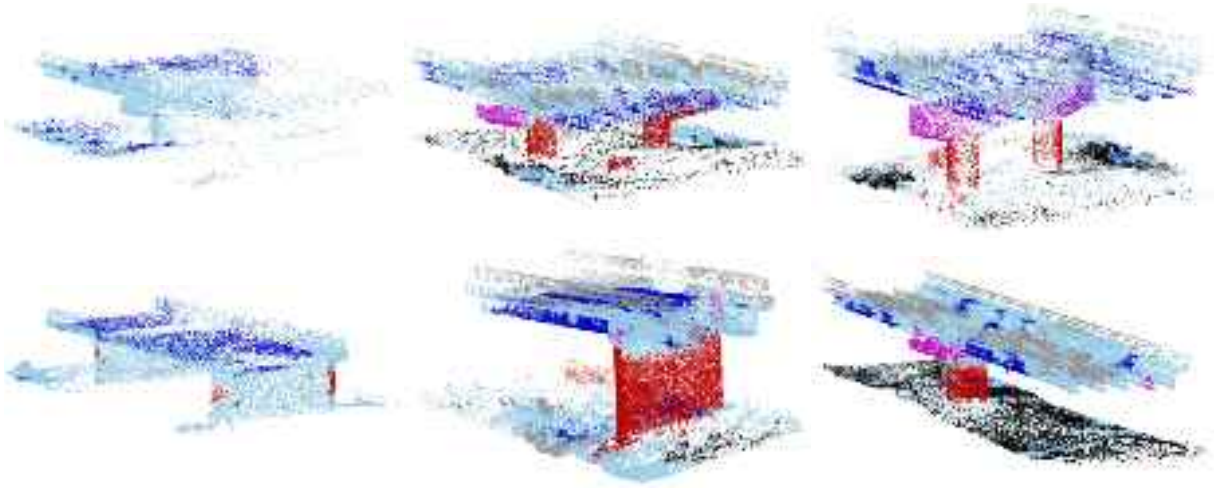


Fig. 13. Dynamic threshold for each class with epoch.



(a) Iteration around epoch 5 (b) Iteration around epoch 30 (c) Iteration around epoch 60

Fig. 14. Visualization of pseudo-labels across training iterations. Each batch consists of a stochastically shuffled subset of real-world bridges. Points with prediction confidence below the threshold (shown in light blue) are excluded from training (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 4
Ablation experiments on the R1_Split2 test set with varying threshold determination methods in ST.

Method	mIoU	Acc	AP	AP_{50}	AP_{25}
Fixed-threshold [30]	32.9	49.2	25.9	45.1	**64.7**
Self-paced Learning Policy [32]	29.2	43.4	16.6	41.2	50.4
CBDT (Proposed)	**41.4**	**61.1**	**34.8**	**52.5**	59.2

Table 5
Overall performance comparison on R1_Split2 test set.

Configuration	mIoU	AP	AP_{50}	AP_{25}
Source-Only (Baseline)	40.4	32.3	49.7	56.1
Mix-UDA (Proposed)	**41.4**	**34.8**	**52.5**	**59.2**
Mix (Upper bound)	81.5	58.1	72.4	84.4
Overall improvement	**+1.0**	**+2.5**	**+2.8**	**+3.1**

The CBDT method enables reliable pseudo-label generation during training, as shown in Fig. 14. In the early training phase, only the pier and deck classes yield high-confidence pseudo-labels. As training progresses, background, piercap, and railing progressively meet their confidence thresholds and are incorporated in training, resulting in increasingly reasonable and trustworthy pseudo-labels.

5.2. *Comparative evaluation of CBDT with existing ST methods*

This section compares the experimental results of the proposed CBDT method, the fixed-threshold approach, and the self-paced learning

Table 6
Class-wise mIoU comparison on R1_Split2 test set.

Configuration	mIoU	Background	Pier	Railing	Deck	Piercap
Source-Only (Baseline)	40.4	**37.9**	63.6	**14.4**	**58.2**	28.1
Mix-UDA (Proposed)	**41.4**	34.8	**66.4**	11.9	52.4	**41.3**
Overall improvement	**+1.0**	−3.1	**+2.8**	−2.5	−5.8	**+13.2**

Table 7
Class-wise AP scores comparison on R1_Split2 test set.

Configuration	AP	Pier	Railing	Deck	Piercap
Source-Only (Baseline)	32.3	**79.0**	0.0	48.9	1.4
Mix-UDA (Proposed)	**34.8**	74.6	0.0	**56.1**	**8.3**
Overall improvement	**+2.5**	−4.4	0.0	**+7.2**	**+6.9**

policy (introduced in CBST) [32] on the R1_Split2 test set, as shown in Table 4. For semantic segmentation, CBDT outperforms the fixed-threshold approach by +8.5 % mIoU and + 11.9 % Acc, and the self-paced learning policy by +12.2 % mIoU and + 17.7 % Acc. In instance segmentation, CBDT shows improvements of +8.9 % in AP and + 7.4 % in AP_{50} over the fixed-threshold approach, and + 18.2 % in AP, +11.3 % in AP_{50}, and + 8.8 % in AP_{25} compared to the self-paced learning policy. While CBDT demonstrates broad performance improvements, particularly at higher IoU thresholds, the AP_{25} metric decreases against the fixed-threshold method, indicating a trade-off at lower IoU thresholds. These results suggest that CBDT better handles data distribution imbalance and excels in high-precision segmentation tasks.

5.3. Quantitative evaluation of CBDT for UDA performance

This section evaluates and discusses the domain adaptation performance of Mix-UDA incorporating CBDT on the R1_Split2 dataset, comparing it with the Source-Only baseline and the Mix upper-bound configuration. As shown in Table 5, the Source-Only model achieved an mIoU of 40.4 % and an AP of 32.3 %. Segmentation accuracy was particularly high for critical bridge components such as piers (mIoU: 63.6 %, AP: 79.0 %) and decks (mIoU: 58.2 %, AP: 48.9 %). It is worth noting that this synthetic dataset was not specifically designed for the real-world R1 test set, but instead represents a more general framework for bridge synthetic data generation (described in Section 3.1). The results indicate that our synthetic dataset enables a certain degree of generalization from synthetic to real-world domains.

The table also shows that the performance of the proposed Mix-UDA approach with CBDT shows improvement from the Source-Only baseline toward the upper-bound performance of the Mix configuration, which uses target-domain ground truth annotations. Compared to the Source-Only baseline, the proposed approach achieved the gains of +1.0 % in mIoU and + 2.5 %, +2.8 %, and + 3.1 % in AP, AP_{25}, and AP_{50}, respectively. In terms of class-wise performance (Tables 6 and 7), the piercap class shows the most significant improvements, with increases of +13.2 % in mIoU and + 6.9 % in AP, while performance on the deck class improves by +7.2 % in AP. Railing recognition remains challenging, primarily due to sparse point cloud density and ambiguous boundaries with the deck. While existing ST methods often struggle with class imbalance and distribution shifts due to unreliable pseudo-labels (Table 4), CBDT exploits correlations between source and target domains to deliver more robust UDA outcomes.

Figs. 15 and 16 visualize the segmentation results for the real-world bridges, where Mix-UDA outperforms the Source-Only model trained on the synthetic dataset, achieving higher accuracy on key components. Fig. 17 highlights its advantages in challenging regions, such as the underside of the deck and the deck–pier and railing–deck transition zones, where Mix-UDA produces more precise semantic boundaries. Such improved performance in the boundary regions enhances instance segmentation performance, preventing the misclassification of transition areas as spurious instances, as seen with the Source-Only results. These findings demonstrate the effectiveness of the proposed domain adaptation approach in improving segmentation performance without the need for manual annotations.

5.4. Quantitative evaluation of CBDT for domain generalization

To explore the generalizability of the network trained with the Mix-UDA configuration, this section evaluates the network performance under domain generalization settings, where the test domains are unseen during training. Specifically, this research trains the network using the Mix-UDA configuration with S1_Split1 as the source domain and R1_Split1 as the target domain, and evaluates the model on three unseen domains: R2, R3, and R4 datasets.

Mix-UDA consistently outperforms the Source-Only baseline across various domains, as shown in Table 8. It improves mIoU by +5.0 % on R2 and + 0.5 % on R3, with corresponding AP gains of +4.4 % and + 1.5 %. Although there is a slight mIoU drop on R4 (−2.1 %), AP increases significantly by +11.1 %, indicating strong structural component detection under challenging domain shifts.

Mix-UDA achieves better overall performance on semantic segmentation, particularly for background and primary structural components, with mIoU increases for background (+5.9 %) and pier (+4.8 %), as shown in Table 9. Despite slight drops in deck and railing performance, overall semantic segmentation robustness is preserved, as illustrated in

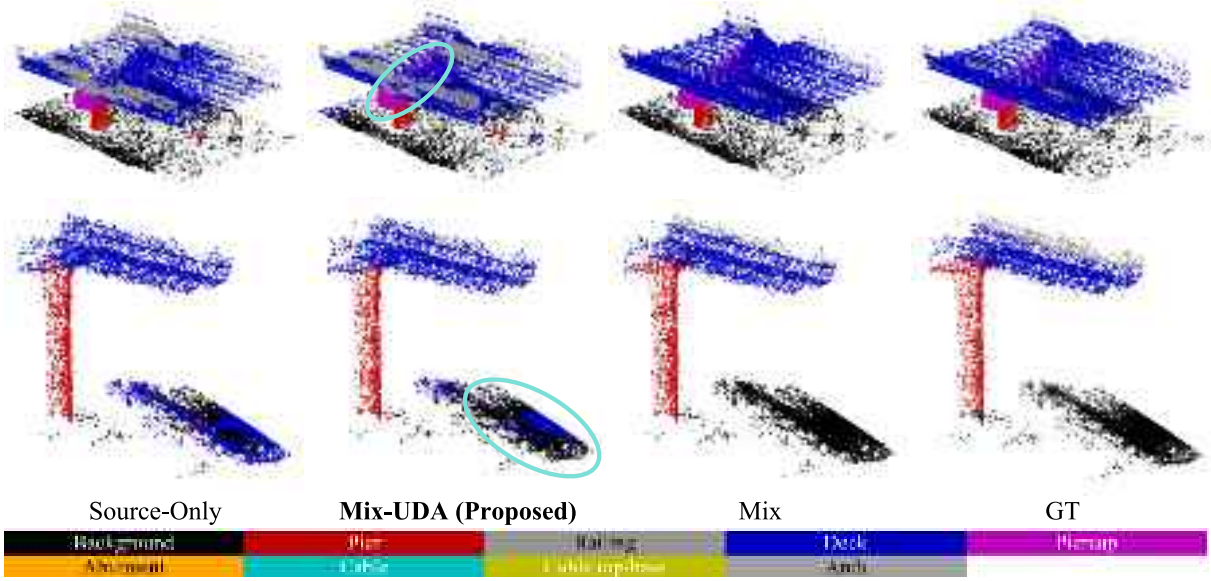


Fig. 15. Visualization of semantic segmentation results on R1_Split2 test set (circle-highlighted key regions).

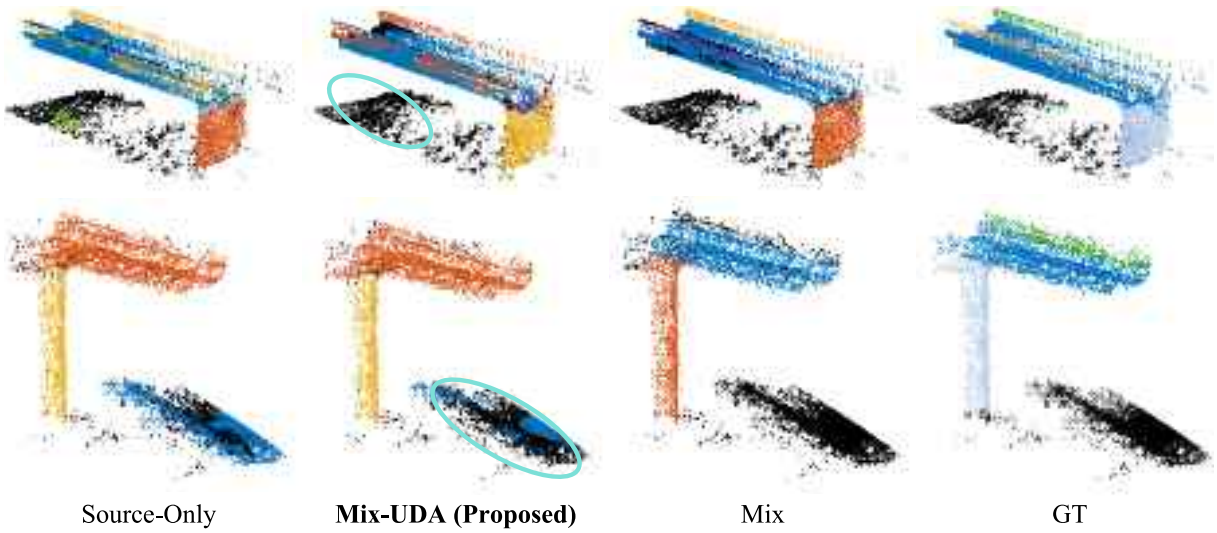


Fig. 16. Visualization of instance segmentation results on R1_Split2 test set (circle-highlighted key regions).

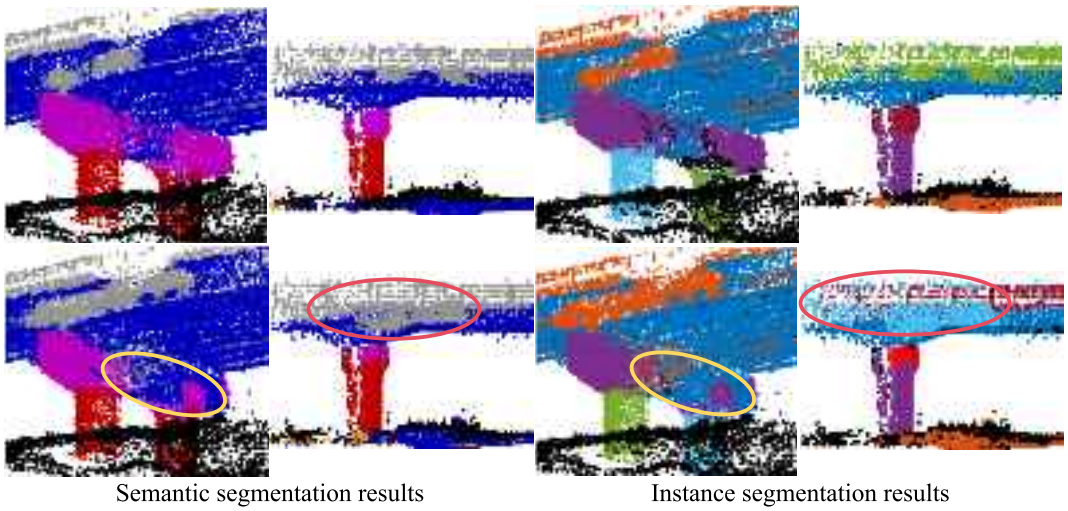


Fig. 17. Performance comparison of ***Mix-UDA (top)*** and *Source-Only (bottom)* segmentation results in the underside of the deck and longitudinal transition zones between components (circle-highlighted key regions).

Table 8
Overall performance comparison on R2-R4_All test sets.

Configuration	R2_All Test		R3_All Test		R4_All Test	
	mIoU	AP	mIoU	AP	mIoU	AP
Source-Only (Baseline)	68.6	36.4	62.8	35.2	**37.4**	25.9
Mix-UDA (Proposed)	**73.6**	**40.8**	**63.3**	**36.7**	35.3	**37.0**
Mix (Upper bound)	83.0	52.7	73.2	40.7	64.0	40.3
Overall improvement	**+5.0**	**+4.4**	**+0.5**	**+1.5**	−2.1	**+11.1**

Table 9
Class-wise mIoU comparison on combined (R2 + R3 + R4)_All test sets.

Configuration	mIoU	Background	Pier	Railing	Deck	Piercap
Source-Only (Baseline)	58.6	74.2	58.8	**64.1**	**86.7**	**9.0**
Mix-UDA (Proposed)	**59.7**	**80.1**	**63.6**	62.8	85.0	7.1
Overall improvement	**+1.1**	**+5.9**	**+4.8**	−1.3	−1.7	−1.9

Fig. 18. Mix-UDA also demonstrates superior capability in instance segmentation (Table 10), particularly for key bridge components, with AP improvements of +18.1 % for pier and + 14.0 % for deck. As shown in Fig. 19, Source-Only and Mix sometimes misclassify the deck as background, whereas Mix-UDA consistently identifies it correctly. These results confirm that the proposed CBDT method contributes to learning robust and transferable feature representations during UDA, leading to consistent improvements over the baseline even under domain generalization scenarios.

This research extends the evaluation of domain adaptation and generalization by training the network with various combinations of source and target domain data. As shown in Table 11, the training set always consists of Split1 of the corresponding domain(s), while validation is consistently performed on Split2 subsets. The results demonstrate that the model trained with S1 as the source domain and R1 as the target domain outperforms the Source-Only baseline in all cases except R3, confirming the observations from previous sections. The diagonal and the first column of the Mix-UDA results in the table show that the CBDT-based UDA approach improves performance across diverse target domains, highlighting its cross-domain effectiveness of the proposed method. Regarding domain generalization (off-diagonal entries

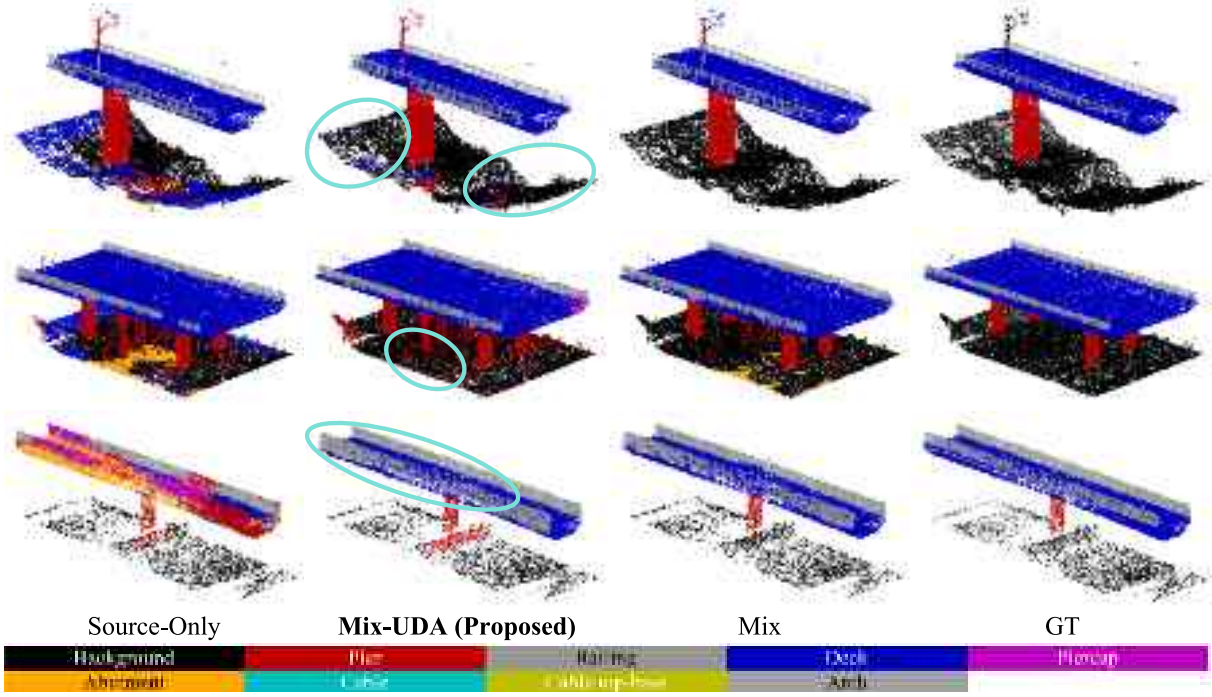


Fig. 18. Visualization of semantic segmentation results on (R2 + R3 + R4)_All test set (circle-highlighted key regions).

Table 10
Class-wise AP scores comparison on combined (R2 + R3 + R4)_All test sets.

Configuration	AP	Pier	Railing	Deck	Piercap
Source-Only (Baseline)	30.0	60.0	**10.0**	40.3	**9.5**
Mix-UDA (Proposed)	**34.4**	**78.1**	5.1	**54.3**	0.0
Overall improvement	**+4.4**	**+18.1**	−4.9	**+14.0**	−9.5

excluding the first column), the model achieves noticeably better results than the baseline when tested on the R4 domain. However, the performance drop on R3 suggests that the generalization ability of the proposed method may vary across domains, influenced by inherent domain characteristics. In the specific experiments conducted in this research, this decline in performance can likely be attributed to the extreme scarcity of bridges with piercaps in R3, resulting in a highly imbalanced class distribution (as shown in Fig. 10). Consequently, the features learned from other target domains may differ significantly from those in R3, leading to the observed performance degradation. Nevertheless, the proposed approach demonstrates superior average performance across all domains compared to the baseline, indicating its robustness and potential. This improvement becomes even more remarkable when excluding the underperforming R3 domain, further underlining the method's potential.

The observed challenges in domain generalization and their underlying causes can be summarized as follows:

(1) Adaptation to highly diverse domains simultaneously: The first column of Table 11 suggests that augmenting target domain data from different domains may not necessarily improve the UDA performance. When substantial distribution differences exist among multiple target domains, the effectiveness of domain adaptation may be limited.
(2) Generalization to significantly different unseen test domains: When test domains differ significantly from the source and target domains considered during training, the learned representations may struggle to generalize to the test domain.
(3) Mitigation of the inherent risks of pseudo-labeling: Despite the robustness introduced in the proposed CBDT method, the method still relies on the assumption that high-confidence prediction regions correspond to correct classes. When the teacher network produces inaccurate predictions in real-world domains, this assumption is violated, leading to pseudo-labeling error accumulation and ultimately resulting in performance degradation.

Possible approaches for addressing these challenges include: (1) developing methods to automatically detect and determine whether a new deployment domain differs substantially from the originally trained target domain, (2) further aligning the composition and characteristics of bridge components by improving the synthetic data generation approach, and (3) identifying and addressing anomalous behaviors in dynamically adjusted thresholds during training.

5.5. *Practical implications or potential applications*

By eliminating the need for time-consuming and costly instance-level annotations of real-world point clouds, this research facilitates the scaling of 3D instance segmentation algorithms to a wider range of bridges, components, and data collection settings. This is achieved through (1) an extensible and scalable synthetic-data generation pipeline that produces annotated datasets more efficiently than collecting and labeling real-world data, and (2) a unified UDA framework that incorporates real-world datasets acquired under diverse equipment, measurement configurations, structural typologies, and environmental conditions. Retraining may still be required if a new domain differs substantially from those used in previous training, depending on its similarity to the existing domains. Appropriate evaluation metrics will be addressed in future work.

The proposed approach establishes a foundational step in Scan-to-BrIM workflows. It provides a basis for automated extraction of component dimensions, surface conditions, and defect information, as well as the construction of geometric models through shape fitting and spatial relationship analysis (e.g., based on Industry Foundation Classes (IFC)). This foundational capability lays the groundwork for more advanced, automated, and reliable methodologies for bridge inspection,

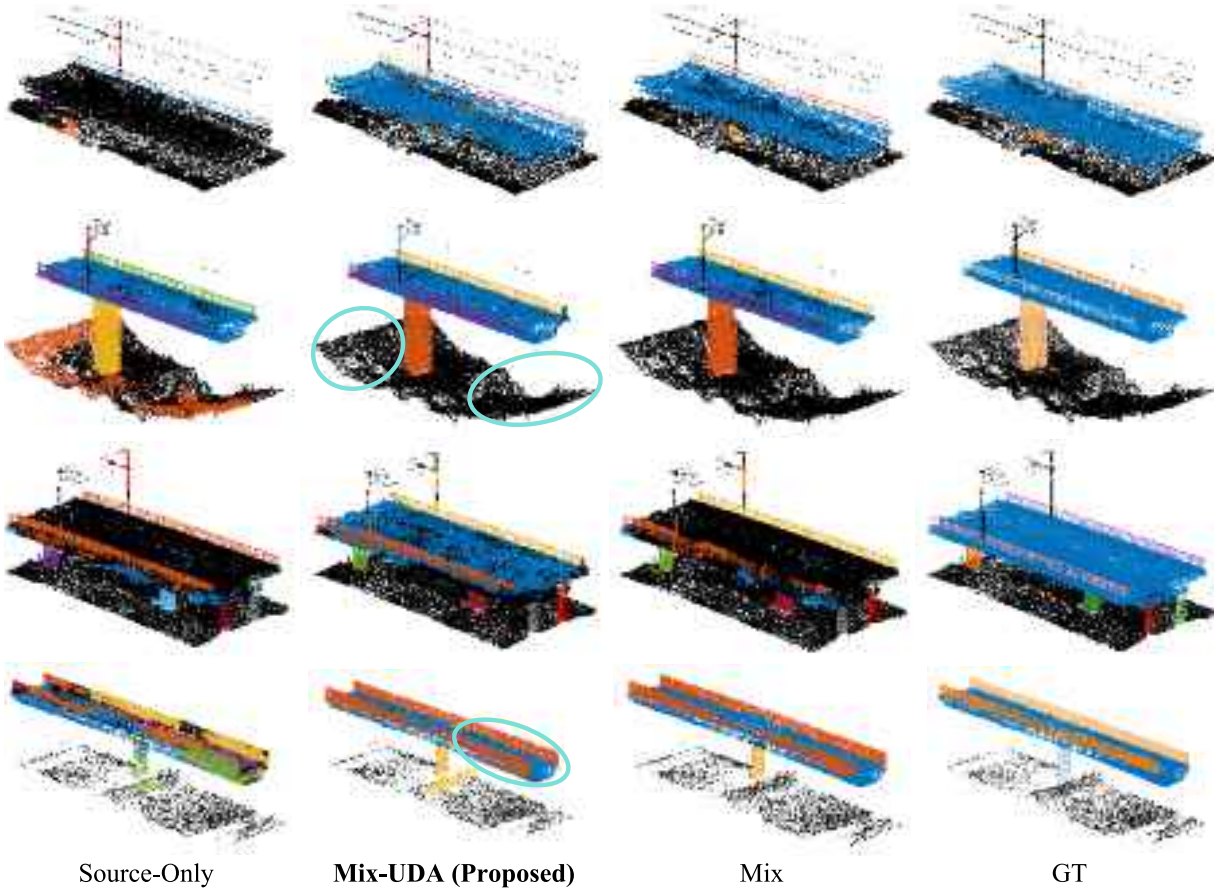


Fig. 19. Visualization of instance segmentation results on (R2 + R3 + R4)_All test set (circle-highlighted key regions).

Table 11
Performance comparison on multi-domain UDA.

Configuration	Target Domain	R1		R2		R3		R4		Avg		Avg (w/o R3)	
		mIoU	AP	mIoU	AP	mIoU	AP	mIoU	AP	mIoU	AP	mIoU	AP
Source-Only (Baseline)	–	40.4	32.3	62.9	37.8	63.3	41.4	33.3	5.9	50.0	29.4	45.5	25.3
Mix-UDA (Proposed)	R1	41.4 (+**1.0**)	34.8 (+**2.5**)	66.0 (+**3.1**)	41.0 (+**3.2**)	61.8 (−1.5)	39.0 (−2.4)	66.0 (+**32.7**)	13.0 (+**7.1**)	58.8 (+**8.8**)	32.0 (+**2.6**)	57.8 (+**12.3**)	29.6 (+**4.3**)
	R1 + R2	25.2 (−15.2)	17.3 (−15.0)	64.1 (+**1.2**)	46.9 (+**9.1**)	62.6 (−0.7)	38.0 (−3.4)	52.1 (+**18.8**)	29.6 (+**23.7**)	51.0 (+**1.0**)	32.9 (+**3.5**)	47.1 (+**1.6**)	31.3 (+**6.0**)
	R1 + R3	45.8 (+**5.4**)	28.2 (−4.1)	59.3 (−3.6)	30.1 (−7.7)	61.1 (−2.2)	33.0 (−8.4)	55.1 (+**21.8**)	32.4 (+**26.5**)	55.3 (+**5.3**)	30.9 (+**1.5**)	53.4 (+**7.9**)	30.2 (+**4.9**)
	R1 + R4	39.4 (−1.0)	33.7 (+**1.4**)	48.9 (−14.0)	26.1 (−11.7)	60.1 (−3.2)	34.4 (−7.0)	57.0 (+**23.7**)	45.4 (+**39.5**)	51.4 (+**1.4**)	34.9 (+**5.5**)	48.4 (+**2.9**)	35.1 (+**9.8**)

Performance comparison on multi-domain UDA across different target domain combinations (R1, R1 + R2, R1 + R3, R1 + R4) on all test domains (R1-R4), including both target and unseen domains. The shaded portions (diagonal and first column) highlight domain adpatation on target domains, while the unshaded portions (off-diagonal) focus on domain generalization, showing performance on unseen domains.

monitoring, and maintenance.

6. Conclusions and future work

This paper developed an approach for performing deep learning-based instance segmentation on bridge point clouds, particularly in the presence of severe class imbalance and distribution shifts common in real-world data. To alleviate the reliance on extensive data collection and manual annotations for diverse real-world bridge management scenarios, this research proposed a comprehensive unsupervised domain adaptation (UDA) approach. The proposed methodology begins with a synthetic point cloud data generation of diverse bridge types with ground-truth instance labels, based on the simulated UAV-based image collection and SfM-based 3D reconstruction. Then, UDA was integrated into a two-stage DL-based instance segmentation algorithm. By applying UDA during the semantic segmentation stage, instance segmentation performance improved without relying on hard-to-acquire pseudo-instance labels. Finally, a ST method, termed class-balanced dynamic thresholding (CBDT), was developed to address the challenge of severe class imbalance and distribution shifts. The CBDT method dynamically adjusts class-wise confidence thresholds for pseudo-labeling by analyzing per-class confidence distributions at each iteration, without relying on predetermined rules whose optimal settings are problem-specific.

Extensive validation was performed on four real-world bridge point cloud datasets, each varying in acquisition methods, structural types,

dimensions, and geographic locations. In each experiment, one real-world dataset was chosen as the target domain for UDA, while the other three were reserved for domain generalization testing. The proposed framework applied UDA for semantic segmentation between the labeled synthetic domain and the unlabeled target domain, with the instance refinement stage trained exclusively on labeled synthetic data. Compared to a baseline trained solely on synthetic data, the proposed method enhanced performance on the target domain, with overall mIoU increasing by +1.0 % (from 40.4 % to 41.4 %), overall AP by +2.5 % (from 32.3 % to 34.8 %), AP_{50} by +2.8 % (from 49.7 % to 52.5 %), and AP_{25} by +3.1 % (from 56.1 % to 59.2 %). The proposed method also demonstrated domain generalization across the three unseen real-world domains, achieving consistent AP improvements for each domain. Notably, it achieved a significant increase of +11.1 % on one domain, while the AP improved by +4.4 % (from 30.0 % to 34.4 %) across the combined dataset of the three domains. These results underscore the robustness of the proposed approach in achieving domain-generalizable component extraction.

Potential directions for further enhancement include: (1) improving the synthetic data generation approach by aligning the composition and characteristics of bridge components with real-world conditions and incorporating advanced physics-based simulations, sensor noise models, and realistic terrain features to narrow the synthetic-to-real domain gap, (2) developing methods to automatically identify and distinguish whether a new deployment domain differs substantially from the originally trained target domain, enabling more adaptive model updating or retraining when necessary, and (3) extending UDA directly to the generation of instance-level pseudo-labels. These improvements are expected to achieve performance comparable to supervised learning using target domain annotations.

Moreover, the current work will be extended to the following applications in the future: (1) extension to additional bridge types, such as arch, cable-stayed, and suspension bridges, (2) automated extraction of segmented component dimensions, surface conditions, and defect information. These extensions are expected to advance the bridge management process with enhanced automation in bridge digital twinning and scan-to-BrIM.

CRediT authorship contribution statement

Jiawei Xu: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Mingyu Shi:** Writing – review & editing, Supervision, Data curation, Conceptualization. **Rafael Cabral:** Writing – review & editing, Supervision, Software, Data curation. **Diogo Ribeiro:** Writing – review & editing, Supervision, Conceptualization. **Weilei Yu:** Writing – review & editing, Supervision, Data curation, Conceptualization. **Huayong Wu:** Supervision, Resources. **Yasutaka Narazaki:** Writing – review & editing, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Yasutaka Narazaki reports financial support was provided by National Natural Science Foundation of China. Huayong Wu reports financial support was provided by Shanghai Municipal Commission of Housing and Urban Rural Development. Rafael Cabral reports financial support was provided by Foundation for Science and Technology. Diogo Ribeiroc reports financial support was provided by Foundation for Science and Technology. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors acknowledge the financial support from the National Science Foundation of China (Grant No. 52361165658), Shanghai Urban Digital Transformation Special Fund Project (202401069), and Scientific Research Program of Shanghai Municipal Commission of Housing and Urban Rural Development (2024-Z02-003). This research is also supported in part by UID/04708 of the CONSTRUCT - Instituto de I&D em Estruturas e Construções, funded by FCT, I.P./ MCTES through the national funds, as well as the grant UI/BD/150970/2021 Portuguese Science Foundation, FCT/MCTES.

Data availability

Data will be made available on request.

References

[1] M. Nasrollahi, G. Washer, Estimating inspection intervals for bridges based on statistical analysis of National Bridge Inventory Data, J. Bridg. Eng. 20 (2015) 04014104, https://doi.org/10.1061/(ASCE)BE.1943-5592.0000710.
[2] B.F. Spencer, V. Hoskere, Y. Narazaki, Advances in computer vision-based civil infrastructure inspection and monitoring, Engineering 5 (2019) 199–222, https://doi.org/10.1016/j.eng.2018.11.030.
[3] A.M. Rakoczy, D. Ribeiro, V. Hoskere, Y. Narazaki, P. Olaszek, W. Karwowski, R. Cabral, Y. Guo, M.M. Futai, P. Milillo, R. Santos, A. Trias, L. Gonzalez, J. C. Matos, F. Schmidt, Technologies and platforms for remote and autonomous bridge inspection–review, Struct. Eng. Int. (2024), https://doi.org/10.1080/10168664.2024.2368220.
[4] Y. Perez-Perez, M. Golparvar-Fard, K. El-Rayes, Scan2BIM-NET: deep learning method for segmentation of point clouds for scan-to-BIM, J. Constr. Eng. Manag. 147 (2021) 04021107, https://doi.org/10.1061/(ASCE)CO.1943-7862.0002132.
[5] T. Xia, J. Yang, L. Chen, Automated semantic segmentation of bridge point cloud based on local descriptor and machine learning, Autom. Constr. 133 (2022) 103992, https://doi.org/10.1016/j.autcon.2021.103992.
[6] D. Lamas, A. Justo, M. Soilán, M. Cabaleiro, B. Riveiro, Instance and semantic segmentation of point clouds of large metallic truss bridges, Autom. Constr. 151 (2023) 104865, https://doi.org/10.1016/j.autcon.2023.104865.
[7] B. Riveiro, M.J. DeJong, B. Conde, Automated processing of large point clouds for structural health monitoring of masonry arch bridges, Autom. Constr. 72 (2016) 258–268, https://doi.org/10.1016/j.autcon.2016.02.009.
[8] R. Lu, I. Brilakis, C.R. Middleton, Detection of structural components in point clouds of existing RC bridges, Comp. Aid. Civil Eng. 34 (2019) 191–212, https://doi.org/10.1111/mice.12407.
[9] Y. Yan, J.F. Hajjar, Automated extraction of structural elements in steel girder bridges from laser point clouds, Autom. Constr. 125 (2021) 103582, https://doi.org/10.1016/j.autcon.2021.103582.
[10] J.S. Lee, J. Park, Y.-M. Ryu, Semantic segmentation of bridge components based on hierarchical point cloud model, Autom. Constr. 130 (2021) 103847, https://doi.org/10.1016/j.autcon.2021.103847.
[11] X. Yang, E. del Rey Castillo, Y. Zou, L. Wotherspoon, Y. Tan, Automated semantic segmentation of bridge components from large-scale point clouds using a weighted superpoint graph, Autom. Constr. 142 (2022) 104519, https://doi.org/10.1016/j.autcon.2022.104519.
[12] B. Yang, J. Wang, R. Clark, Q. Hu, S. Wang, A. Markham, N. Trigoni, Learning object bounding boxes for 3D instance segmentation on point clouds, in: Advances in Neural Information Processing Systems 32, 2019, pp. 6740–6749, https://doi.org/10.48550/arXiv.1906.01140.
[13] J. Hou, A. Dai, M. Nießner, 3D-SIS: 3D semantic instance segmentation of RGB-D scans, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 4416–4425, https://doi.org/10.1109/CVPR.2019.00455.
[14] F. Engelmann, M. Bokeloh, A. Fathi, B. Leibe, M. Nießner, 3D-MPA: multi-proposal aggregation for 3D semantic instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 9028–9037, https://doi.org/10.1109/CVPR42600.2020.00905.
[15] D.D.T. Tran, B. Kang, Y. Lee, MSTA3D: multi-scale twin-attention for 3D instance segmentation, in: Proceedings of the 32nd ACM International Conference on Multimedia, 2024, pp. 1467–1475, https://doi.org/10.1145/3664647.3680667.
[16] S. Chen, J. Fang, Q. Zhang, W. Liu, X. Wang, Hierarchical aggregation for 3D instance segmentation, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 15447–15456, https://doi.org/10.1109/ICCV48922.2021.01518.
[17] L. Jiang, H. Zhao, S. Shi, S. Liu, C.-W. Fu, J. Jia, PointGroup: dual-set point grouping for 3D instance segmentation, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 4866–4875, https://doi.org/10.1109/CVPR42600.2020.00492.
[18] T. Vu, K. Kim, T.M. Luu, T. Nguyen, C.D. Yoo, SoftGroup for 3D instance segmentation on point clouds, in: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, New Orleans, LA, USA, 2022, pp. 2698–2707, https://doi.org/10.1109/CVPR52688.2022.00273.

[19] T. Vu, K. Kim, T. Nguyen, T.M. Luu, J. Kim, C.D. Yoo, Scalable SoftGroup for 3D instance segmentation on point clouds, IEEE Trans. Pattern Anal. Mach. Intell. 46 (2024) 1981–1995, https://doi.org/10.1109/TPAMI.2023.3326189.
[20] M. Kolodiazhnyi, A. Vorontsova, A. Konushin, D. Rukhovich, OneFormer3D: one transformer for unified point cloud segmentation, in: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024, pp. 20943–20953, https://doi.org/10.1109/CVPR52733.2024.01979.
[21] Y. Zhao, G. Cong, J. Shi, C. Miao, QueryFormer: a tree transformer model for query plan representation, in: Proceedings of the VLDB Endowment, 2022, pp. 1658–1670, https://doi.org/10.14778/3529337.3529349.
[22] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, B. Leibe, Mask3D: mask transformer for 3D semantic instance segmentation, in: 2023 IEEE International Conference on Robotics and Automation (ICRA), 2023, pp. 8216–8223, https://doi.org/10.1109/ICRA48891.2023.10160590.
[23] Y. Jing, B. Sheil, S. Acikgoz, Segmentation of large-scale masonry arch bridge point clouds with a synthetic simulator and the BridgeNet neural network, Autom. Constr. 142 (2022) 104459, https://doi.org/10.1016/j.autcon.2022.104459.
[24] D. Lamas, A. Justo, M. Soilán, B. Riveiro, Automated production of synthetic point clouds of truss bridges for semantic and instance segmentation using deep learning models, Autom. Constr. 158 (2024) 105176, https://doi.org/10.1016/j.autcon.2023.105176.
[25] A.U. Rahman, V. Hoskere, Instance segmentation of reinforced concrete bridge point clouds with transformers trained exclusively on synthetic data, Autom. Constr. 173 (2025) 106067, https://doi.org/10.1016/j.autcon.2025.106067.
[26] M. Shi, H. Kim, Y. Narazaki, Development of large-scale synthetic 3D point cloud datasets for vision-based bridge structural condition assessment, Adv. Struct. Eng. 27 (2024) 2901–2928, https://doi.org/10.1177/13694332241260077.
[27] Y. Ganin, V. Lempitsky, Unsupervised domain adaptation by backpropagation, in: Proceedings of the 32nd International Conference on Machine Learning, PMLR, 2015, pp. 1180–1189, https://doi.org/10.48550/arXiv.1409.7495.
[28] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, J.W. Vaughan, A theory of learning from different domains, Mach. Learn. 79 (2010) 151–175, https://doi.org/10.1007/s10994-009-5152-4.
[29] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (2010) 1345–1359, https://doi.org/10.1109/TKDE.2009.191.
[30] L. Hoyer, D. Dai, L. Van Gool, DAFormer: improving network architectures and training strategies for domain-adaptive semantic segmentation, in: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022, pp. 9914–9925, https://doi.org/10.1109/CVPR52688.2022.00969.
[31] W. Tranheden, V. Olsson, J. Pinto, L. Svensson, DACS: domain adaptation via cross-domain mixed sampling, in: 2021 IEEE Winter Conference on Applications of Computer Vision (WACV), 2021, pp. 1378–1388, https://doi.org/10.1109/WACV48630.2021.00142.
[32] Y. Zou, Z. Yu, B.V.K.V. Kumar, J. Wang, Unsupervised domain adaptation for semantic segmentation via class-balanced self-training, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 289–305, https://doi.org/10.1007/978-3-030-01219-9_18.
[33] Y. Narazaki, W. Pang, G. Wang, W. Chai, Unsupervised domain adaptation approach for vision-based semantic understanding of bridge inspection scenes without manual annotations, J. Bridg. Eng. 29 (2024) 04023118, https://doi.org/10.1061/JBENF2.BEENG-6490.
[34] J. Yang, S. Shi, Z. Wang, H. Li, X. Qi, ST3D++: Denoised self-training for unsupervised domain adaptation on 3D object detection, IEEE Trans. Pattern Anal. Mach. Intell. 45 (2023) 6354–6371, https://doi.org/10.1109/TPAMI.2022.3216606.
[35] H. Cheng, W. Chai, J. Hu, W. Ruan, M. Shi, H. Kim, Y. Cao, Y. Narazaki, Random bridge generator as a platform for developing computer vision-based structural inspection algorithms, J. Infrastruct. Intellig. Resili. 3 (2024) 100098, https://doi.org/10.1016/j.iintel.2024.100098.
[36] Y. Narazaki, V. Hoskere, K. Yoshida, B.F. Spencer, Y. Fujino, Synthetic environments for vision-based structural condition assessment of Japanese high-speed railway viaducts, Mech. Syst. Signal Process. 160 (2021), https://doi.org/10.1016/j.ymssp.2021.107850.
[37] J. Hoffman, D. Wang, F. Yu, T. Darrell, FCNs in the wild: pixel-level adversarial and constraint-based, Adaptation (2016), https://doi.org/10.48550/arXiv.1612.02649.
[38] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, M. Chandraker, Learning to adapt structured output space for semantic segmentation, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 7472–7481, https://doi.org/10.1109/CVPR.2018.00780.
[39] H. Wang, T. Shen, W. Zhang, L.-Y. Duan, T. Mei, Classes matter: a fine-grained adversarial approach to cross-domain semantic segmentation, in: Computer Vision – ECCV 2020, Springer International Publishing, Cham, 2020, pp. 642–659, https://doi.org/10.1007/978-3-030-58568-6_38.
[40] P. Zhang, B. Zhang, T. Zhang, D. Chen, Y. Wang, F. Wen, Prototypical Pseudo label Denoising and target structure learning for domain adaptive semantic segmentation, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021, pp. 12409–12419, https://doi.org/10.1109/CVPR46437.2021.01223.
[41] N. Zhang, W.H. Mahmoud, Lidar point cloud semantic segmentation using SqueezeSegV2 deep learning network, in: 2025 17th International Conference on Advanced Computational Intelligence (ICACI), 2025, pp. 22–29, https://doi.org/10.1109/ICACI65340.2025.11096294.
[42] D. Rozenberszki, O. Litany, A. Dai, UnScene3D: unsupervised 3D instance segmentation for indoor scenes, in: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024, pp. 19957–19967, https://doi.org/10.1109/CVPR52733.2024.01886.
[43] vvoovv, Import of Google 3D Cities, GitHub, 2024. https://github.com/vvoovv/blosm/wiki/Import-of-Google-3D-Cities (accessed August 14, 2025).
[44] Y. Narazaki, V. Hoskere, G. Chowdhary, B.F. Spencer, Vision-based navigation planning for autonomous post-earthquake inspection of reinforced concrete railway viaducts using unmanned aerial vehicles, Autom. Constr. 137 (2022), https://doi.org/10.1016/j.autcon.2022.104214.
[45] A. Dai, A.X. Chang, M. Savva, M. Halber, T. Funkhouser, M. Nießner, ScanNet: richly-annotated 3D reconstructions of indoor scenes, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2432–2443, https://doi.org/10.1109/CVPR.2017.261.
[46] I. Armeni, O. Sener, A.R. Zamir, H. Jiang, I. Brilakis, M. Fischer, S. Savarese, 3D semantic parsing of large-scale indoor spaces, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 1534–1543, https://doi.org/10.1109/CVPR.2016.170.
[47] M. Chen, Q. Hu, Z. Yu, H. Thomas, A. Feng, Y. Hou, K. McCullough, F. Ren, L. Soibelman, STPLS3D: A Large-Scale Synthetic and Real Aerial Photogrammetry 3D Point Cloud Dataset, 2022, https://doi.org/10.48550/arXiv.2203.09065.
[48] O. Ronneberger, P. Fischer, T. Brox, U-net: Convolutional networks for biomedical image segmentation, in: Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Springer International Publishing, Cham, 2015, pp. 234–241, https://doi.org/10.1007/978-3-319-24574-4_28.
[49] B. Graham, M. Engelcke, L. van der Maaten, 3D semantic segmentation with submanifold sparse convolutional networks, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 9224–9232, https://doi.org/10.1109/CVPR.2018.00961.
[50] A. Tarvainen, Harri Valpola, Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results, in: Advances in Neural Information Processing Systems 30, 2017, pp. 1195–1204, https://doi.org/10.48550/arXiv.1703.01780.
[51] H. Kim, J. Yoon, S. Sim, Automated bridge component recognition from point clouds using deep learning, Struct. Control. Health Monit. 27 (2020), https://doi.org/10.1002/stc.2591.
[52] H. Kim, Y. Narazaki, B.F. Spencer Jr., Automated bridge component recognition using close-range images from unmanned aerial vehicles, Eng. Struct. 274 (2023) 115184, https://doi.org/10.1016/j.engstruct.2022.115184.
[53] R. Cabral, R. Oliveira, D. Ribeiro, A.M. Rakoczy, R. Santos, M. Azenha, J. Correia, Railway bridge geometry assessment supported by cutting-edge reality capture technologies and 3D as-designed models, Infrastructures 8 (2023) 114, https://doi.org/10.3390/infrastructures8070114.
[54] R. Cabral, R. Santos, J. Correia, D. Ribeiro, Optimal reconstruction of railway bridges using a machine learning framework based on UAV photogrammetry and LiDAR, Struct. Infrastruct. Eng. (2025) 1–21, https://doi.org/10.1080/15732479.2025.2531562.
[55] National Bridge Inspection Standards - Bridge Inspection - Safety Inspection - Bridges & Structures, Federal Highway Administration, 2004. https://www.fhwa.dot.gov/bridge/nbis.cfm (accessed May 4, 2025).
[56] CloudCompare - Open Source Project. https://www.cloudcompare.org/, 2009 (accessed August 27, 2025).
[57] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 3431–3440, https://doi.org/10.1109/CVPR.2015.7298965.
[58] K. He, G. Gkioxari, P. Dollár, R. Girshick, Mask R-CNN, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980–2988, https://doi.org/10.1109/ICCV.2017.322.
[59] D.P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, 2017, https://doi.org/10.48550/arXiv.1412.6980.
[60] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, The cityscapes dataset for semantic urban scene understanding, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3213–3223, https://doi.org/10.1109/CVPR.2016.350.