Automation in Construction 142 (2022) 104459

Contents lists available at ScienceDirect

Automation in Construction

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

Segmentation of large-scale masonry arch bridge point clouds with a synthetic simulator and the BridgeNet neural network

Yixiong Jing *, Brian Sheil, Sinan Acikgoz

Department of Engineering Science, University of Oxford, Parks Road, Oxford OX1 3PJ, UK

ARTICLE INFO

Keywords:
Masonry arch bridge
Deep learning
BridgeNet
Synthetic dataset
Semantic segmentation
Geometric feature extraction

ABSTRACT

Masonry arch bridges constitute the majority of the European bridge stock and feature a wide range of geometric characteristics. Due to a general lack of construction drawings, their geometry is difficult to parameterize. Laser scanning devices are commonly used to capture bridge geometry. However, this requires time-consuming segmentation of point clouds into their constituent components to extract key geometric parameters. To increase efficiency, a 3D deep learning neural network called BridgeNet is proposed that can automate the segmentation. To tackle the scarcity of labelled point cloud data, a synthetic dataset is created to train BridgeNet. BridgeNet is subsequently tested on real point clouds and achieves state-of-art performance, demonstrating the utility of synthetic training data and the advantages of the new network design. Segmented components are then fitted with primitive shapes by using Random Sample Consensus based algorithms to characterize key geometric parameters to assist assessments and inspections.

1. Introduction

Around 60% of the railway bridge stock in Europe is constructed of masonry. These bridges were built during the 19th and early 20th centuries and still play an important role in the railway network. In the UK, there are approximately 18,000 operational masonry railway bridges [1]. These bridges feature ageing materials and are subjected to increasing traffic demands. It is therefore necessary to inspect and assess their safety regularly [2]. These activities require detailed knowledge of the geometry of the structure. However, construction drawings are often unavailable for masonry arch bridges.

Laser scanning is an efficient and non-contact technology to capture the in-situ 3D geometry of large-scale civil engineering infrastructure with points clouds [3]. The obtained point clouds can help assessments by providing information on bridge geometry, such as the radius of curvature and generatrix direction of the arch, which are difficult to measure in the field. Parametric [4] and non-parametric [5,6] shape fitting methods for point cloud components can be used to automate the identification of such geometric parameters.

Point clouds can also be used for inspecting geometric distortions and defects in structural components. Ye et al. [7] have developed algorithms to fit segmented arch and pier point clouds with idealized 3D cylinders and 2D planes to reveal geometric distortions introduced by past settlements. Other studies [8,9] used irregularities in surface normal vectors, roughness and colour to identify and classify defects such as cracking and material loss. Separately, point clouds have been used to quantify the size of structural defects [10] and track deformations occurring over time [11–13].

These previous studies demonstrate the potential for point clouds to help assess, inspect and monitor masonry arch bridges. However, the large size of typical point clouds often requires significant amounts of time for semantic and instance segmentation of constituent components. To recognise the potential of laser scanning in structural engineering practice, it is necessary to develop automated segmentation algorithms which can be used by practising engineers and asset managers.

Semantic and instance segmentation methods adopted in the literature can be roughly classified into three categories, which are feature-based, machine learning and deep learning methods. Feature-based methods consider infrastructure components as primitive shapes organized according to an idealized topology. Segmentation is then conducted based on these assumptions [14,15]. Machine learning methods consist of unsupervised methods which cluster points with similar geometric features followed by supervised learning methods that classify the clustered points [16,17]. Although these methods can achieve remarkable segmentation accuracy, the algorithms are designed for specific topologies with hand crafted geometric features, that are

* Corresponding author.
E-mail addresses: yixiong.jing@wolfson.ox.ac.uk (Y. Jing), brian.sheil@eng.ox.ac.uk (B. Sheil), sinan.acikgoz@eng.ox.ac.uk (S. Acikgoz).

https://doi.org/10.1016/j.autcon.2022.104459
Received 7 April 2022; Received in revised form 30 May 2022; Accepted 23 June 2022
Available online 16 July 2022

difficult to generalize to other cases. In contrast, deep learning (DL) based segmentation techniques do not require the prior definition of features since they can be learned latently by the neural network. Previous applications of DL to the segmentation of heritage buildings [18,19] and bridge point clouds [20] have demonstrated competitive performance relative to established machine learning methods.

There are two significant impediments to using DL techniques to segment masonry arch bridge point clouds: (i) lack of a sufficiently large and labelled point cloud dataset, which is often required for achieving good segmentation accuracy and (ii) lack of neural networks for conducting efficient semantic segmentation on large-scale point clouds.

To tackle challenge (i), researchers in the autonomous driving field augment real datasets with synthetic data obtained from LiDAR simulators [21–24]. The mixed (real+synthetic) training dataset enables significant performance improvement compared to training neural networks on the real dataset alone [25]. In the infrastructure domain, Ma et al. [26] and Morbidoni et al. [27] train the network with synthetic data and show its potential to address data scarcity (see also Narazaki et al. [28]).

Recent research efforts using point-based DL architectures show some promise in tackling the challenge (ii). Voxel [29–35] and graph-based [36–40] neural networks are memory-inefficient since raw point clouds are transformed into other data representations. In contrast, point-based methods [41–51] can take raw point clouds as direct input. However, pioneering point-based architectures like PointNet++ [42] are memory-intensive and unsuitable for large-scale point clouds due to the complicated design of local feature extractors. The recently proposed RandLA-Net [50] can perform semantic segmentation on millions of points in real-time by adopting a lightweight architecture. However, this network can demonstrate poor performance in segmentation tasks which require the neural network to be aware of complex local features. FG-Net [51] applies a feature condensing mechanism to alleviate memory consumption while managing to retain local information. As a consequence, FG-Net [51] outperforms other semantic segmentation neural networks on the public dataset Shape-Net [52]. To address instance segmentation within the DL framework, the similarity matrix [53–55] and a top-down hierarchical structure [56] have been proposed. However, these approaches require heavy computations and are not suitable for large-scale point clouds. Instead, a memory-efficient clustering machine learning method called DBSCAN [57] may be used to perform instance segmentation.

To enable the application of DL to bridge segmentation, this paper develops a synthetic masonry arch bridge simulator that can overcome the data scarcity challenge. In addition, the paper develops a computationally efficient semantic and instance segmentation pipeline to segment large-scale point clouds. Fig. 1 demonstrates the workflow adopted in this paper. To this end, Section 2 describes the development of a new neural network called BridgeNet for semantic segmentation. Section 3 discusses the formulation of synthetic data used to train the network and introduced the real point cloud data used for testing. Section 4 evaluates BridgeNet semantic segmentation accuracy and compares it with state-of-art techniques. Section 5 discusses post-processing methods for efficient instance segmentation and the extraction of useful geometric parameters from instances for engineering purposes.

2. BridgeNet

As shown in Fig. 2, the architecture of BridgeNet adopts the popular encoder-decoder structure to effectively extract features of point clouds at different resolutions. The input size of our neural network is $N \times C_{input}$ in which N is the number of points and C_{input} is the input channel size; C_{input}= 3 since we only provide the raw x, y and z point coordinates as input. This input is first sent to a 1 × 1 convolution layer to extract the feature representations of each point. It is then passed through four Residual Feature Encoder (Residual FE) layers to learn local features in different resolutions. We apply random sampling (RS) to downsample point clouds with a ratio of 4, to achieve time efficiency for large-scale point clouds [50]. The downsampling ratio is adopted from PointNet++ [42] which retains 25% of points at each step. RS was preferred to the more commonly used furthest point sampling as it prevented overfitting. The downsampled features are sent to a Global Feature Extractor which is adapted from [51] to enhance the global point-wise relationship. Four decoder layers are used to scale the number of points back to the original input by using the nearest interpolation method [42]. For each upsampling layer in the decoder, interpolated features are concatenated with the corresponding encoding layer features shown by orange arrows. This is to enhance representations of feature patterns. The output features from the decoder are then passed through a shared Multi Layer Perceptron (MLP) followed by a dropout layer to alleviate overfitting [50]. Three different drop ratios (0.2, 0.5 and 0.8) were tested, in which the test accuracy for 0.5 outperformed 0.2 by a small margin while 0.8 was shown to cause underfitting. Therefore, a drop ratio of 0.5 is used, consistent with other neural networks such as RandLA-Net [50]. Finally, a 1 × 1 convolution layer is used to learn the semantic labels. The Residual FE and Global Feature Extractor [51] blocks are explained in the following sections.

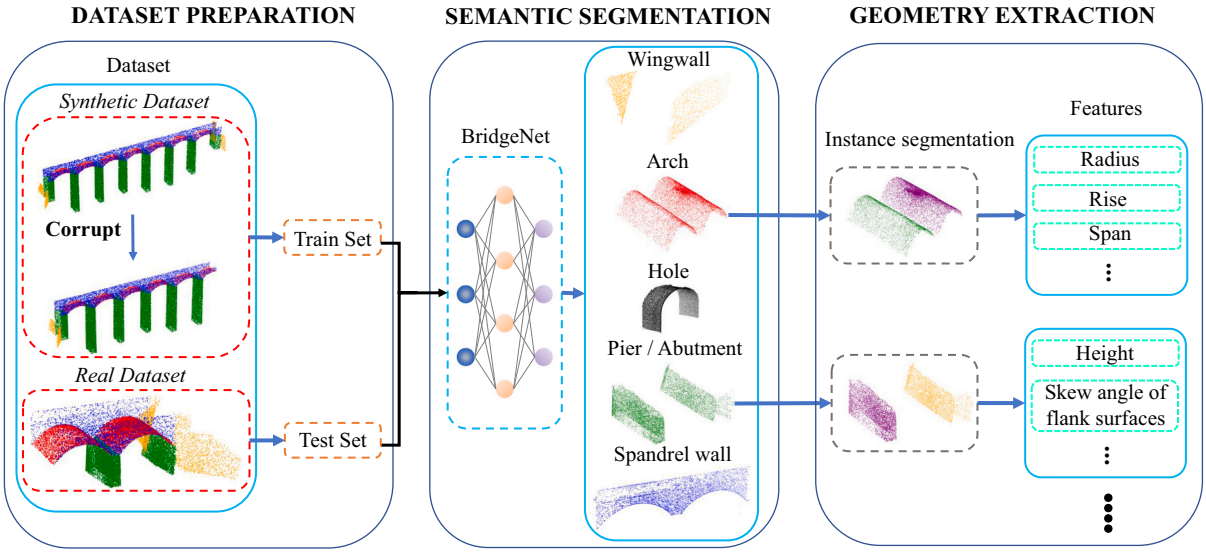


Fig. 1. Research pipeline for automating segmentation of masonry arch bridge point clouds for geometry parameterization.

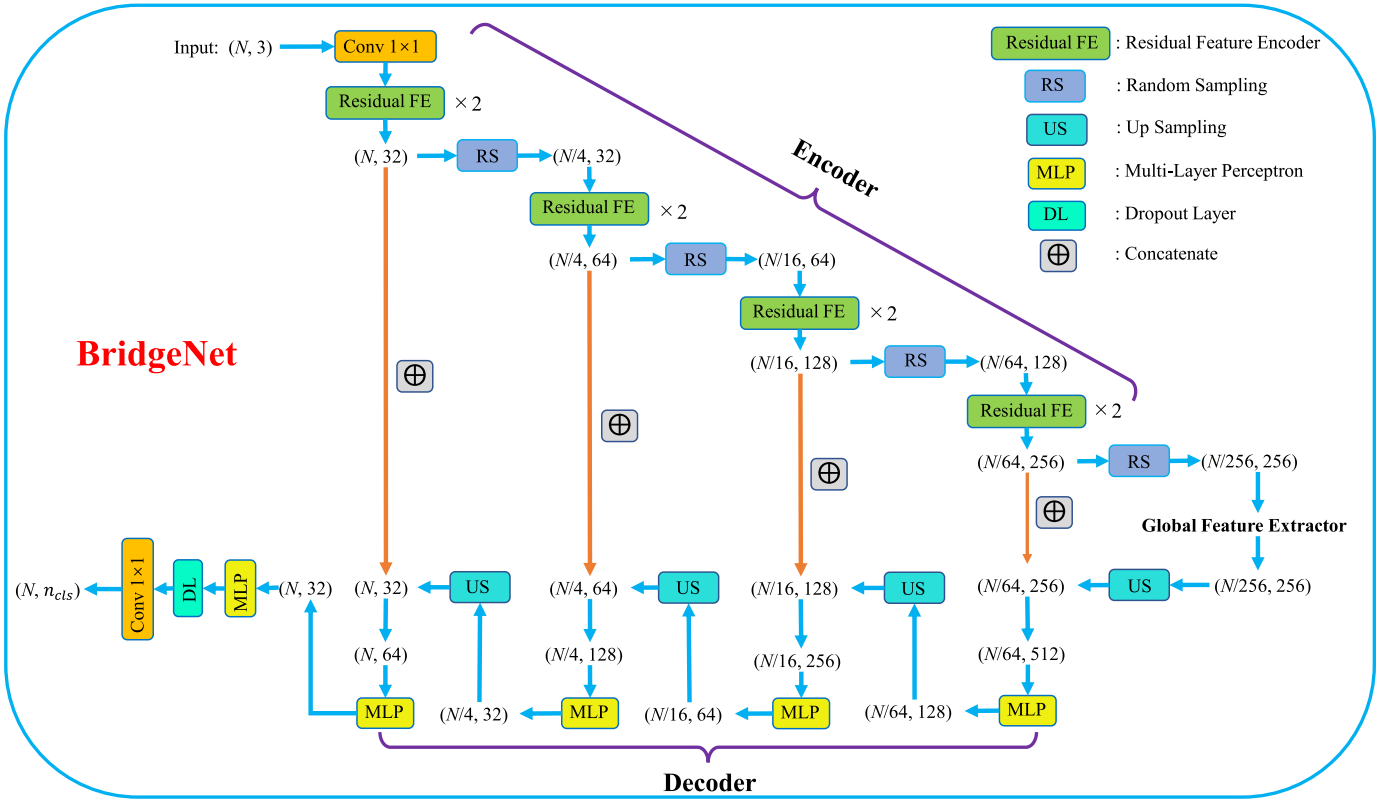


Fig. 2. Overview of BridgeNet. The input point cloud has dimensions $(N,3)$, in which N is the point number and 3 represents the x, y, z coordinates. Features are extracted by the encoder which contains four Residual Feature Encoder (Residual FE) layers and downsampled after each Residual FE layer by using RS. Then a Global Feature Extractor is used to correlate pairwise global relationships. Finally, a decoder upsamples the point cloud back to the original point size using interpolations and convolutional layers to provide the output with dimension (N, n_{cls}), in which n_{cls} stands for semantic labels.

2.1. Residual Feature Encoder

The original local feature aggregation module from [50] was not designed for learning complex local features. Therefore, as shown in Fig. 3, the local feature aggregation module [50] was redesigned to enable it to capture local characteristics. The module contains four basic components: (1) Feature Grouping and Downsampling, (2) Geometric Grouping and Feature Balance, (3) Local Convolution, and (4) Selective Attention, which are explained as follows:

(1) *Feature Grouping and Downsampling:* Let p_i be the x, y and z coordinate of a given point within a point cloud $\boldsymbol{P}$ in which $p_i \in \mathbb{R}^3$.

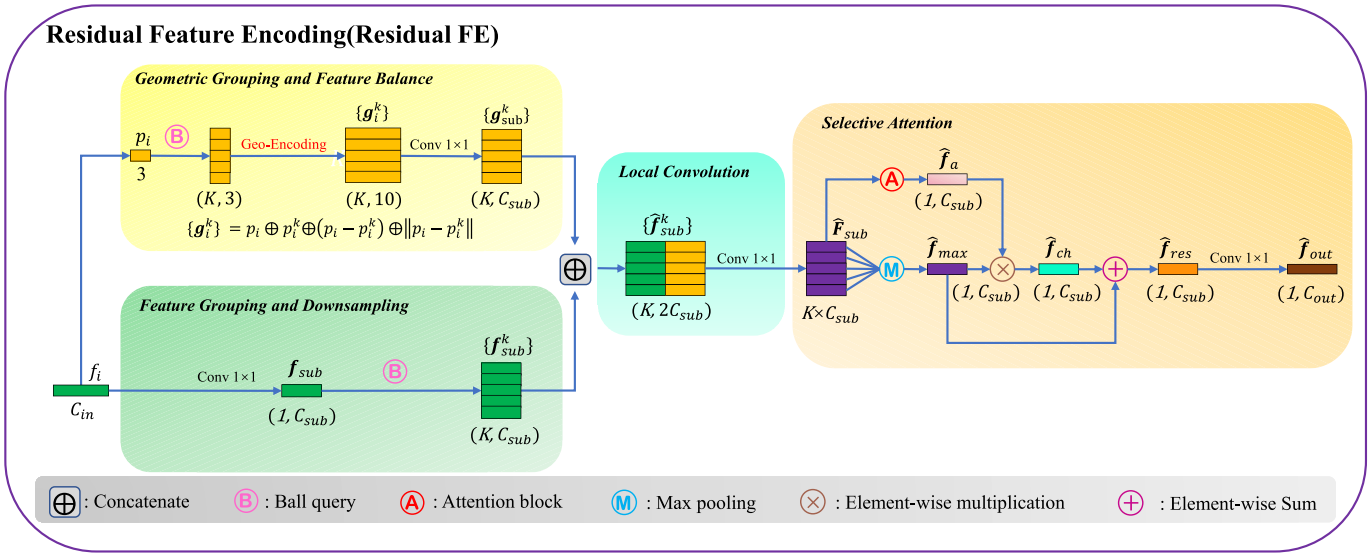


Fig. 3. The modules in the Residual Feature Encoding (Residual FE) layer. The learned latent features are downsampled before applying ball query to save memory in Feature Grouping and Downsampling module. Geometric Grouping and Feature Balance explicitly encodes and balances grouped geometric features in feature dimensions.

The corresponding feature vector for p_i is denoted f_i, where $f_i \in \mathbb{R}^{C_{in}}$. For all points belonging to point cloud $\boldsymbol{P}$, features are first downgraded from C_{in} to $C_{sub} = C_{in}/M$ by a 1×1 convolution before the application of neighbour query to save memory [51]. $M = 8$ is adopted here similar to FG-Net [51], in order to achieve computational efficiency. According to [51], a larger M would be expected to cause a reduction in accuracy while a smaller M would be expected to lead to an increase in the GPU memory consumption. Ball query is then applied to gather the K nearest neighbouring points within a sphere of radius r in which the k-th neighbour is mathematically defined as $\{p_i^k \in \mathbb{R}^3, \|p_i - p_i^k\| \leq r\}$, where $k = 1, 2, 3, \ldots, K$. The local feature $\{\boldsymbol{f}_{sub}^k\}$ is then derived by stacking all sub-sampled neighbour features together.

(2) *Geometric Grouping and Feature Balance:* The relative point position encoding is applied to explicitly provide geometric features to assist the network to achieve better performance. This procedure also compensates for the lost information by downgrading feature vectors in the Feature Grouping and Downsampling module. More details can be found in RandLA-Net [50]. The feature dimension of relative point position $\{\boldsymbol{g}_i^k\}$ is balanced by a 1×1 convolution from 10 to C_{sub} to derive $\{\boldsymbol{g}_{sub}^k\}$ which makes the feature dimension equivalent to the local feature $\{\boldsymbol{f}_{sub}^k\}$.

(3) *Local Convolution:* Different from FG-Net [51], after concatenating the grouped features from Feature Grouping and Downsampling and Geometric Grouping and Feature Balance layers, $\left\{\widehat{\boldsymbol{f}}_{sub}^k\right\}$ is encoded through a 1×1 convolution to generate local features $\widehat{\boldsymbol{F}}_{sub}$. It was observed that this step improves the overall performance significantly by explicitly learning from local structures. It also requires considerable memory but this is compensated by feature downsampling steps. Alternative convolutional kernels like KPConv [45] were also tested but these led to severe overfitting.

(4) *Selective Attention:* A self-attention mechanism is applied to weight $\widehat{\boldsymbol{F}}_{sub}$ with respect to their importance in the feature dimension. The channel-wise weighted feature $\widehat{\boldsymbol{f}}_a$ is learned from grouped features which represent the importance of each feature channel from a statistical perspective and are defined as:

$$\widehat{\boldsymbol{f}}_a = softmax(\boldsymbol{w}_{att}\widehat{\boldsymbol{F}}_{sub}) \tag{1}$$

where $\boldsymbol{w}_{att} \in \mathbb{R}^K$ is the attentive score which is learnable for aggregation and *softmax* is a function that normalizes all K scores in each channel so that they sum to 1. Even though both [50,51] adopt summation instead of maxpooling to avoid information loss, it was observed in our dataset that the summation operation can lead to severe overfitting since it groups relatively irrelevant information. Therefore, maxpooling was applied here to select the largest feature from $\widehat{\boldsymbol{F}}_{sub}$ in each channel. The selected feature vector $\widehat{\boldsymbol{f}}_{max}$ is multiplied element-wisely with $\widehat{\boldsymbol{f}}_a$ and a residual connection is used to add the original feature $\widehat{\boldsymbol{f}}_{max}$ to the learned feature $\widehat{\boldsymbol{f}}_{ch}$. Another 1×1 convolution is used in the final step to recover the feature dimension C_{sub} of the residual feature $\widehat{\boldsymbol{f}}_{res}$ to C_{out} of the output $\widehat{\boldsymbol{f}}_{out}$.

2.2. Global Feature Extraction

Global Feature Extraction is proven to be effective in the ablation test of FG-Net [51]. The same structure is also adopted in our network to exploit and enhance globally the relationship of closely related points. As shown in Fig. 4, the input feature $\boldsymbol{F}_{in} \in \mathbb{R}^{N_i \times C}$ ($N_i = N/C$ and $C = 256$ in our case) from the encoder layer is first transformed by two 1×1 convolution layers into latent representations $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$. The matrix multiplications of $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$, followed by normalization, give the pointwise similarity map which is further multiplied pair-wisely with input features to capture related regions more effectively. Finally, the local feature map $\boldsymbol{F}_{in}$ is summed element-wise with learned global contextual representation $\boldsymbol{F}_g$ to derive both locally and globally relation-aware features $\boldsymbol{F}_{out}$. More details can be found in [51].

3. Dataset

The dataset is composed of real and synthetic components, which are used for test and training purposes, respectively.

3.1. Real dataset

The real dataset comprises the point clouds of seven multi-span masonry arch railway bridges from the UK. Before segmentation, these point clouds need to be pre-processed to remove parts that are not modelled by the synthetic dataset, such as the ground and the environment surrounding the bridge. Some bridge components, such as the parapet, were not of interest and were also removed. Poorly scanned areas, which were not intended for detailed analysis were also deleted. Finally, bridge components were manually segmented into 5 classes which are the spandrel wall, pier and abutment, arch, hole (only in the

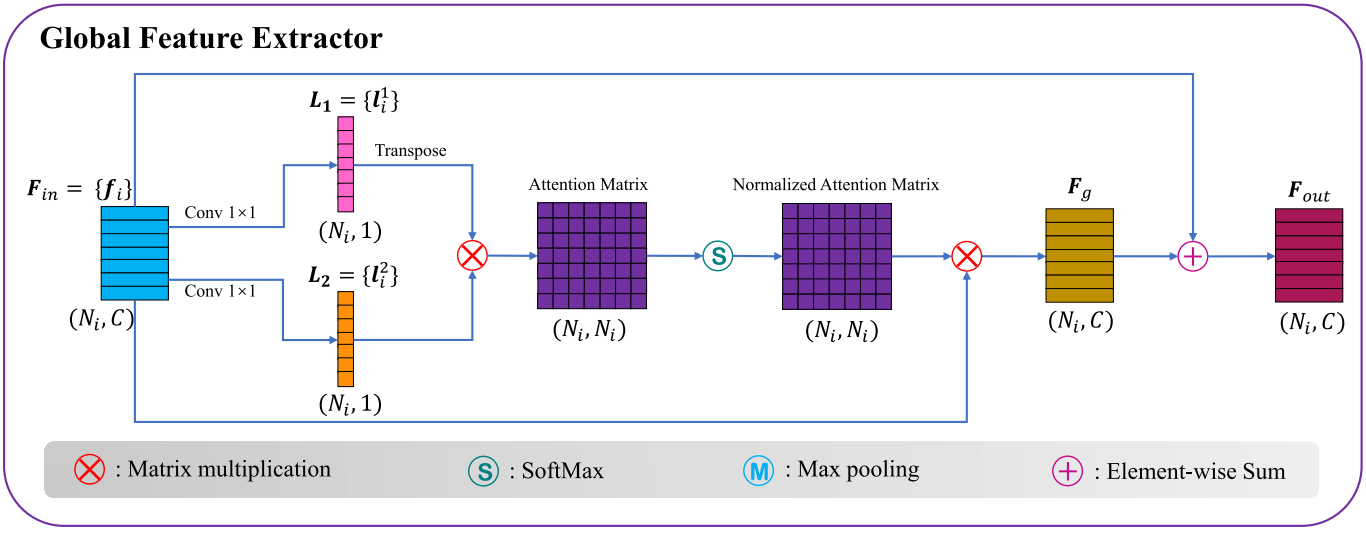


Fig. 4. Graphical explanation of Global Feature Extractor shown in Fig. 2. The global pairwise points relationship is learned and represented by the normalized attention matrix.

Marsh Lane viaduct), and wingwall (only in the Stapleton Road viaduct). Fig. 5 demonstrates this procedure. The real dataset contains the point cloud of 7 masonry railway bridges as shown in Fig. 6. Marsh Lane and Stapleton Road viaduct data did not feature colour information and were therefore visualized with black points in Fig. 6. Each point was classified as a member of one of the 5 following classes:

Spandrel wall: The two side surfaces of the bridge superstructure are known as the spandrel wall. The lower spandrel wall boundary is defined at the top of the piers at the arch 'springing'. Points in this class are shown in blue in Fig. 5.

Pier and abutment: The piers are the columns beneath the bridge superstructure. They can be tapered (see Digswell viaduct in Fig. 6) and can also have a non-rectangular footprint due to track curvature (see Marsh Lane and Chelmsford viaducts). Bridges with a tapering pier are listed in Table 1. Abutments are located at the end of a bridge. They share similar geometric features to piers and are therefore classified in the same category. Piers and abutments are shown in green in Fig. 5.

Arch: Arches are shown in red points and their shape can typically be approximated by a partial cylinder. Elliptical arches exist, but Brencich et al. [58] note that they are rare. Only square spans are considered in this study.

Hole: Holes exist in piers to minimize the use of materials. These holes penetrate the pier in the longitudinal direction of the bridge and are often symmetric with respect to the vertical centerline of the pier. The holes only exist in the scanning of the Marsh Lane viaduct as shown in Fig. 6.

Wingwall: Wingwalls attach to the side surfaces of the abutments. The wingwalls of the Stapleton Road viaduct, shown in Fig. 6, have a triangular shape on one side and a pentagon on the other side. The shape of the wingwall varies drastically according to the local topography of the embankment.

The basic geometric information and the classes included in each bridge are summarised in Table 1.

3.2. Synthetic dataset

The synthetic dataset aims to approximate the geometric properties of two types of real masonry arch bridges: straight and curved bridges (see Table 1). A bottom-to-top methodology is developed to automate synthetic bridge geometry generation by assembling entire bridges from two primitive shapes. As shown in Fig. 7, (1) the two primitive shapes in level 1 are quadrilaterals and partial cylinders; (2) four subassemblies, namely the span, abutment, pier and wingwall, are then derived from the combinations of primitive geometries; (3) different 3D masonry bridges are generated by iteratively producing those components based on bridge topologies. The noiseless synthetic point cloud produced this way is then corrupted randomly to simulate geometric distortions and laser scanning errors that are commonly encountered in practice.

Since the mathematical operations involved in Level 1 and Level 3 (Fig. 7) are well-established, the following sections discuss the definition of the subassembly data, the range of parameters used to specify the size of components, and the corruption of the resulting geometries.

3.2.1. Component generator

The 3D drawings of the span, pier, abutment and wingwall subassemblies are shown in Level 2 of Fig. 7.

Span subassembly: The span subassembly can be considered as the combination of one arch surface and two spandrel wall surfaces. The arch is expressed by a segmental cylinder, for which the range is defined by the minimum and maximum circular angles. The spandrel wall can be defined by first generating a quadrilateral and then filtering the part under the arch.

Pier subassembly: The pier subassembly includes a part of the superstructure that connects the neighbouring spans. It can be defined in the same way as in the span subassembly but without filtering. The lower part of the pier is composed of four quadrilateral surfaces. Two surfaces are coplanar with the spandrel walls and the other two surfaces have a taper angle α relative to the ground. The pier sometimes features a hole that can be built by filtering areas inside the pier that intersect with two partial cylinders sandwiching a rectangular prism.

Abutment subassembly: The abutment subassembly can be represented by a slightly modified version of the pier subassembly.

Wingwall subassembly: The wingwall subassembly takes a variety of shapes. For simplicity, this study represents the wingwall with a 3D right-angled trapezoid. Since the rear surface in contact with the soil is not visible, only the front, top and outward side surfaces are generated by quadrilaterals.

3.2.2. Bridge parameter definition

The range of parameters adopted to describe masonry arch railway bridges refers to the statistical data collected in Italy [58] and the UK [59]. Due to the limited information available, some of the parameters had to be defined empirically. Following Fig. 7, the parameters were defined to describe the four subassemblies:

(1) *Span subassembly:* The number of spans is assumed to range from 5 to 20. The top and front views in Table 2 indicate parameters to define each span subassembly. The parameters include the clear length of the span l_{span}, bridge width w_{span}, rise of the arch s and the fill depth h_{fill_depth}. The synthetic bridge generator uses the specified range for the length of the span l_{span} and determines s and h_{fill_depth} accordingly, following the ratios presented in Table 2. All values are sampled uniformly within the specified parameter ranges. Single track railway bridges were assumed to have a width ranging from 3.5 m to 5 m, and double track bridges, a width of 7 m to 10 m. Either of the two cases has a 50% possibility of being sampled, as indicated in the third column of Table 2.

(2) *Pier subassembly:* Parameters are needed to describe the overall pier geometry and the position and size of the hole. Table 3 specifies the thickness t_{pier}, height h_{pier} and width w_{pier} of the pier. The vertical distance between the crown of the hole to the upper boundary of piers h_{top}, the corresponding bottom distance h_{bottom}, taper angle α, width of the hole m, the rise of the upper and lower arch for the hole s_1 and s_2 are the parameters used to determine the geometry of the hole. The hole

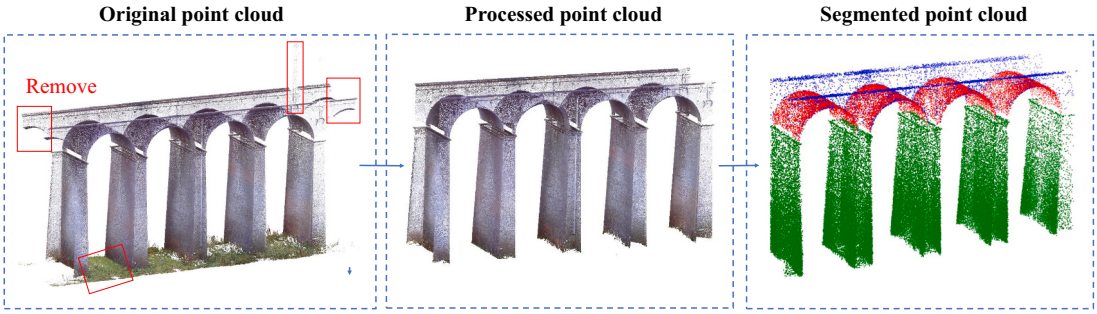


Fig. 5. Graphical representation of real point cloud processing including pre-processing (e.g. removal of irrelevant environment and components) and manual segmentation.

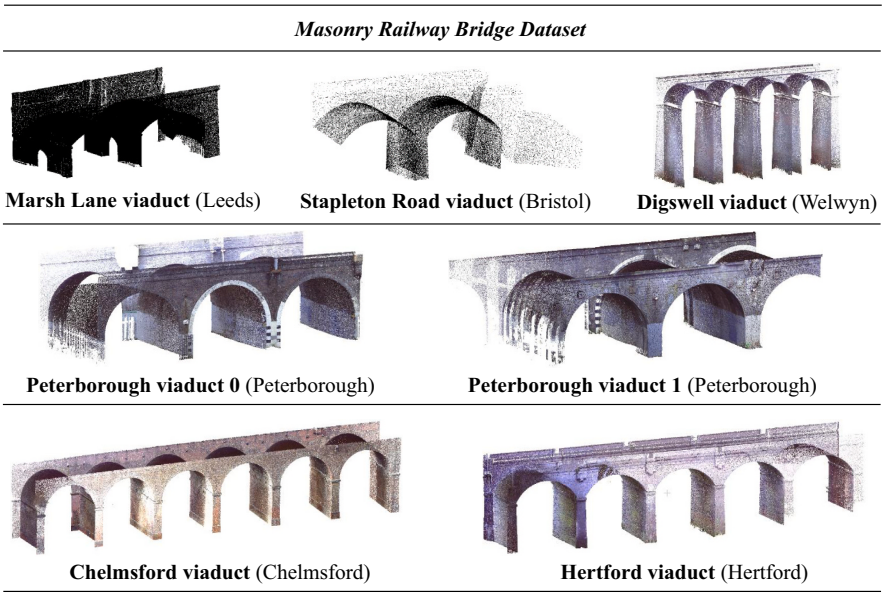


Fig. 6. Real point clouds of seven masonry bridges that constitute in the test dataset.

Table 1
Characteristics of the real point cloud dataset.

				Classes				
		Span	Pier taper	Spandrel	Pier	Arch	Wingwall	Hole
Curved bridge	Chelmsford viaduct	6	X	✓	✓	✓	X	X
	Marsh Lane viaduct	2	X	✓	✓	✓	X	✓
Straight bridge	Stapleton Road viaduct	2	✓	✓	✓	✓	✓	X
	Digswell viaduct	4	✓	✓	✓	✓	X	X
	Peterborough viaduct 0	3	X	✓	✓	✓	X	X
	Peterborough viaduct 1	3	X	✓	✓	✓	X	X
	Hertford viaduct	5	✓	✓	✓	✓	X	X

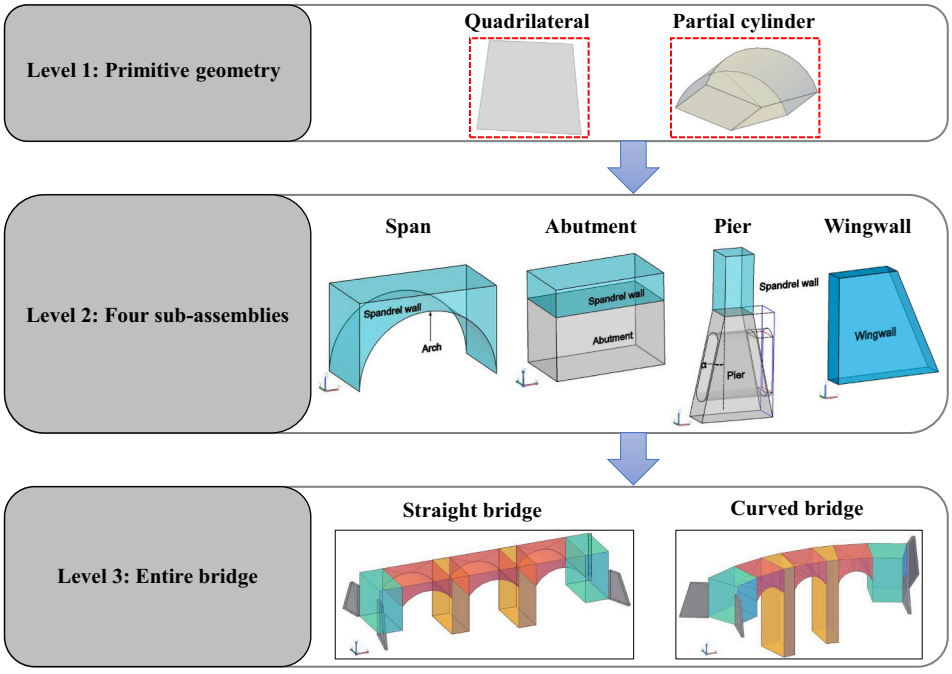
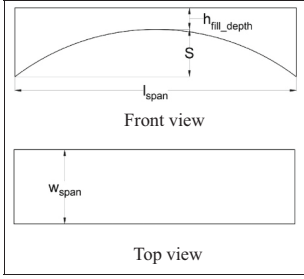


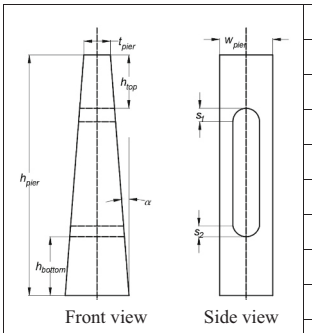
Fig. 7. Flowchart illustrating the generation of synthetic bridges hierarchically from primitive geometry to the complete bridge.

Table 2
Schematic descriptions and the range of geometric parameters used to define span subassemblies.



Span			
Parameter	Range	Probability	Reference
l_{span}	[8 m, 15 m]	70%	[58]
	[15 m, 26 m]	30%	
w_{span}	[3.5 m, 5 m] for one lane	50%	
	[7 m, 10 m] for two lanes	50%	
s/l_{span}	[0.05, 0.50]	100%	
h_{fill_depth}/l_{span}	[0.02, 0.14]	100%	

Table 3
Schematic descriptions and the range of geometric parameters used to define pier subassemblies and the hole.



Pier			
Parameter	Range	Probability	Reference
t_{pier}/l_{span}	[0.1, 0.25]	100%	Empirical and [59]
h_{pier}	[5 m, 20 m]		Empirical
α	[0°, 30°]		
m/w_{pier}	[0.1, 0.4]		
s_1/m	(0, 0.5]		
s_2/m	(0, 0.5]		
h_{top}/h_{pier}	[0.15, 0.35]		
h_{bottom}/h_{pier}	[0.15, 0.35]		

Fig. 8. Example of a pier hole at Stanway Bridge, UK, where the upper and lower arch surfaces are partial cylinders sandwiching a rectangular prism.

geometry is not covered by either [58] or [59], therefore it was defined empirically by the authors based on example bridges (see Fig. 8).

(3) *Abutment subassembly:* The width of abutments is assumed to be the same as w_{span} for geometric consistency. The length of the abutment is also not defined in the literature. Therefore, the height h_{abut} and the length l_{abut}, are defined empirically. The range of the abutment length is defined from [0.2*l_{span}, 0.5*l_{span}] which is thicker than the pier since it is responsible for dispersing the loading to the supporting soil. The height of abutments is defined in the range from 0.3*h_{pier} to h_{pier} (see Fig. 9).

(4) *Wingwall:* 4 wingwalls are attached to the lateral surfaces of the abutments. These are assumed to be 3D trapezoids. From Fig. 10, the length of the upper-line h_{upper}, lower-line h_{lower}, the thickness of the wingwall t_{wing} and the height of the wingwall h_{wing} need to be determined. Due to limited information on the geometry of the wingwalls, the parameter ranges used in the synthetic simulator were based on the authors' experience. h_{upper} is sampled from a uniform distribution with the range of [2 m, 4 m] and h_{lower} with [4 m, 8 m]. h_{wing} is sampled from 0.3*(h_{pier} + h_{span}) to (h_{pier} + h_{span}). The angle between the wingwall and the side surfaces of the abutment is sampled from [−60°, 60°]. t_{wing} is defined to be in the range of 0.25 m to 0.75 m.

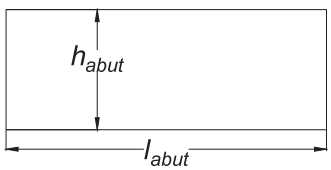
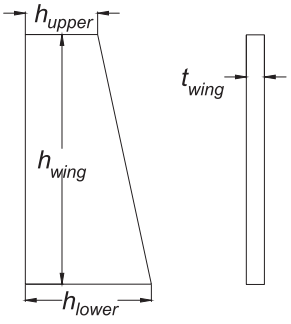


Fig. 9. Top view of the geometric parameters used to define abutments.



(a) Side view (b) Front view

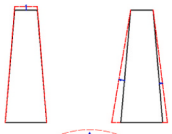
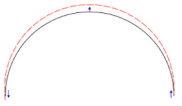
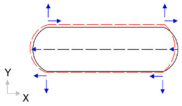
Fig. 10. Schematic descriptions (a), (b) of geometric parameters used to define wingwalls in the synthetic data.

3.2.3. Data corruption and truncation

Geometric irregularities are common in masonry arch bridges. Spans may be of different lengths, pier top elevations may vary across the bridge due to topographical reasons and piers and holes may feature distorted geometries due to support settlements. To simulate these aspects, the geometric properties of the subassemblies need to be randomised, as shown in Table 4. For instance, according to the first row, the span length specified for the bridge will vary uniformly in the range [−0.2 m, 0.2 m] in synthetic point clouds.

Measurement imperfections in point clouds occur due to measurement errors and occlusions. Laser scanning measurement noise was approximated simply by adding a zero-mean Gaussian noise distribution with a standard deviation of 0.005 m to all x, y and z coordinates [23]. The occlusions in real point clouds are case-specific and impossible to capture without detailed models of scanning geometry and the surrounding environment. To approximately capture the incomplete information that can arise as a result of occlusions, data truncation is performed. The synthetic point cloud is truncated from both ends as shown in the plan view in Fig. 11a. The truncation ratio for the left and right parts does not exceed 50% of the bridge length. Since the Marsh Lane viaduct used in the testing dataset contains a truncated hole, some piers were also truncated, as shown in Fig. 11b. This involves first setting a threshold sampled from the uniform distribution with the range of [0.15*h_{pier},0.5* h_{pier}] and subsequently filtering out the points with z coordinates lower than the threshold.

Table 4
Schematic description of the data corruption introduced to each component of the synthetic bridge point clouds. The deformations are indicated by red dotted lines which depart from black lines that represent the initially generated regular bridge geometry

	Component	Randomness	Range (Uniform distribution)
	Front view of a span and a pier	Length	[−0.2 m, 0.2 m]
	Pier	Height and skew angle of flank surfaces	Height: [−0.1*h_{pier}, 0.1*h_{pier}] Skew angle: [−0.5°, 0.5°]
	Arch	Rise and elevations for two ends of the arc	Rise: [−0.1 m, 0.1 m] Elevation: [−0.1 m, −0.1 m]
	Hole	Rise of both lower and upper arches and two directions for four corner points	Rise: [−0.02 m, 0.02 m] X axis: [−0.005 m, 0.005 m] Y axis: [−0.005 m, 0.005 m]

4. Training results

4.1. Implementation details

Our framework is implemented using Pytorch. Adam is used as an optimizer with an initial learning rate of 10^{-3} and a decay rate of 10^{-4}. The x, y and z coordinates of point clouds are first normalized within the range [−1,1]. The coordinates are then randomly rotated around the z-axis with an angle between 0 to 2π. The normalized point cloud is also rescaled with a scalar between 0.8 and 1.25 and shifted from 0 to 0.1 relative to the x, y and z directions for data augmentation. The batch size is 4 for training so the training data has the size (4, 65536, 3), where 4 represents the batch size and 65536 is the number of points for each batch with 3 referring to the x, y, and z coordinates of each point.

The training dataset contains only synthetic data in which up to 2000 point clouds are sampled by using the data generator introduced in Section 3. To be compatible with the real dataset, the ratio between the number of synthetic curved and straight bridges is set to 2/5. The test dataset includes all 7 real bridges. Since most of the test point clouds have millions of points while the default number of input points in the network is 65536, each real bridge point cloud data is randomly sampled into non-repeated sets with the default number of points. For example, the Hertford viaduct point cloud contains 5 million points which can be split into 5000000/65536≈76 sets. The residual points which are not included in the sets are discarded. 70 epochs are trained for each case. The experiments are conducted on an RTX 3080 GPU with 10GB of memory.

4.2. Evaluation metrics

Three different metrics are adopted in the experiments to assess BridgeNet performance and benchmark against existing networks PointNet++ and RandLA-Net. The overall accuracy is defined as:

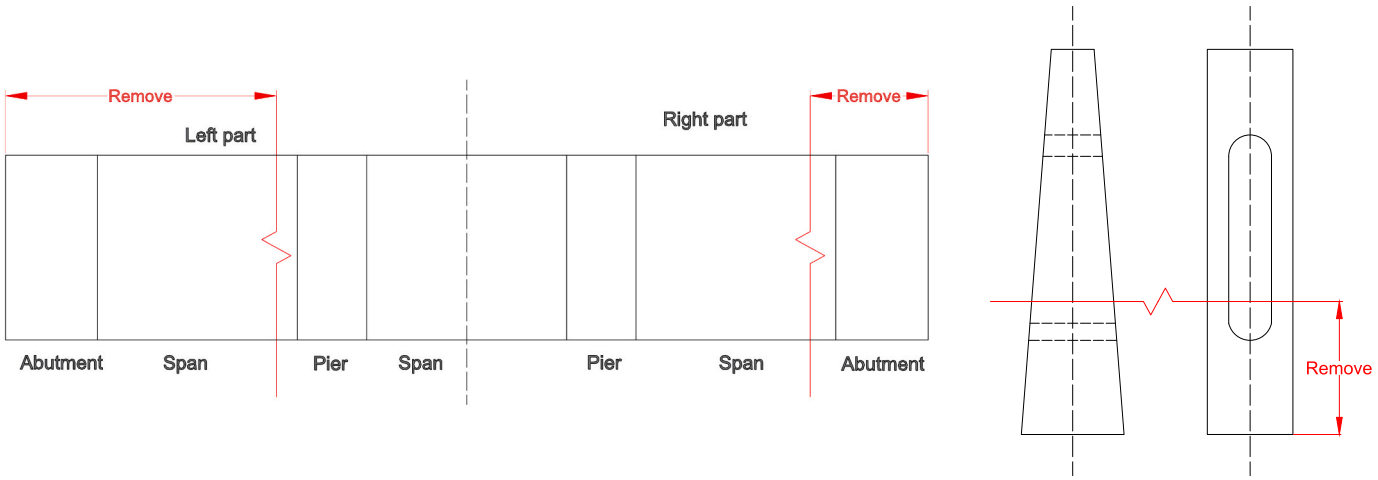
$$Accuracy = TP/N \tag{2}$$

in which N is the number of points and TP (true positive) represents the points that are correctly labelled. Eq. (2) gives overall measurements of the neural network performance though it cannot reflect the segmentation accuracy for each class. Therefore, the mean precision, denoted by *mean_Pr*, and mean intersection over union, *mIoU*, metrics are also used. These are defined in Eqs. (3) and (4),

$$mean\_Pr = \sum_{i=1}^{n_{cls}} (TP_i/(TP_i + FP_i)) \Big/ n_{cls} \tag{3}$$

$$mIoU = \sum_{i=1}^{n_{cls}} (TP_i/(TP_i + FP_i + FN_i)) \Big/ n_{cls} \tag{4}$$

where TP_i is the true positive for class i, FP_i (false positive) is the number of points being misclassified as class i and FN_i (false negative) is the number of misclassified points belonging to class i. Eqs. (3) and (4) can highlight poor segmentation of a certain class even when the class occupies a small proportion of the entire point clouds. In particular, the metric *mIoU* presents the most rigorous evaluation of accuracy, as it also considers the influence of misclassified points FN_i. If the total number of classes is under 5, which is possible since some bridges do not contain hole or wingwall classes (see Table 1), the n_{cls} is adjusted to remove non-existing classes.



(a) Bridge top view (b) Pier front and side

Fig. 11. Data truncation in synthetic bridge models. The first example (a) shows the plan view of a bridge, whose point cloud is truncated from two sides (see zigzagged red lines), while the second example (b) shows the side view of a pier, which is truncated along its height. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

4.3. Semantic segmentation results

Networks are trained on the purely synthetic dataset with an increasing number of bridge point clouds; 100, 400, 500, 600, 1000, 1500 and 2000 point clouds are considered. Once trained, the networks are tested on the real bridge dataset containing the 7 bridges. To ensure like-for-like comparisons, the sample ratio, number of neighbours, input and output channel for each feature extraction block, and number of feature extractors used in RandLA-Net and BridgeNet are set to be the same.

Table 5 presents the evaluation metrics obtained for PointNet++, RandLA-Net and BridgeNet. The best results for each metric are marked with bold numerals and underlined. BridgeNet outperforms the other two networks by a large margin except for the case of 100 point clouds. BridgeNet achieves the best results with 2000 point clouds in the training dataset. As the training data size is increased, the *mIoU* improves steadily, indicating that the synthetic dataset provides valuable new information that helps to improve the performance of the network in segmenting real point clouds. To better understand how BridgeNet performs on real bridge point clouds, the *mIoU* for each component is shown in Table 6. Values of 0 mean that the point cloud does not contain the corresponding class. Both Peter 0 and Peter 1 are poorly classified with average *mIoU* of 0.762 and 0.836, mainly due to the poor classifications of the pier class. For the wingwall and hole, the network achieves satisfactory performance for the wingwall class by achieving a *mIoU* value of 0.704. However, the hole is not well segmented with a *mIoU* value of 0.513.

Fig. 12 visualizes the segmentation results and the ground truth labelling of each bridge. Bridge components could be identified correctly, despite significant differences in bridge geometry and point cloud density amongst the real dataset examples. The misclassifications are highlighted with inset figures, showing close-ups of the misclassified areas. The piers of Peter 0 and Peter 1 on the leftmost side are misclassified with an arch class, likely due to the low density of points in this region. Similar misclassification is observed for other test cases near the arch springing points, indicating that the edge between the objects is not well captured by the neural network. This could be due to the downsampled size of the point cloud, which is used for evaluation. From Fig. 12, it can be seen that the hole in the Marsh Lane viaduct is partially misclassified as a pier element. Nonetheless, despite its truncated shape, the outlines of the hole feature can still be identified. According to Table 6, the wingwall class has a relatively low average class *mIoU* compared with the arch, pier and spandrel wall classes. This discrepancy arises as synthetic point clouds do not model the curved transition between the abutment and the wingwall. Adapting more detailed geometry in synthetic models, or including real point clouds with such features in the training dataset, is likely to improve accuracy further.

Table 5
Comparison of the performance of PointNet++, RandLA-Net and BridgeNet for semantic segmentation on test dataset with increasing training samples in the synthetic dataset using accuracy, mean precision and mean intersection over union metrics.

		Number of synthetic bridges used for training						
		100	400	500	600	1000	1500	2000
Accuracy	PointNet++	0.874	0.915	0.913	0.912	0.895	0.908	0.896
	RandLA-Net	0.845	0.911	0.914	0.911	0.932	0.920	0.927
	BridgeNet	0.875	0.938	0.947	0.941	0.951	0.954	**0.957**
mean_Pr	PointNet++	0.545	0.605	0.641	0.645	0.618	0.623	0.627
	RandLA-Net	0.568	0.678	0.665	0.695	0.666	0.716	0.713
	BridgeNet	0.532	0.685	0.743	0.752	0.785	0.861	**0.867**
mIoU	PointNet++	0.466	0.544	0.586	0.582	0.513	0.564	0.552
	RandLA-Net	0.438	0.562	0.574	0.581	0.583	0.585	0.598
	BridgeNet	0.461	0.629	0.654	0.680	0.684	0.751	**0.779**

Table 6
Mean intersection over union (*mIoU*) of each semantic class for each real bridge.

	Chelmsford	Digswell	Hertford	Peter 0	Peter 1	Stapleton	Marsh Lane
Spandrel	0.837	0.855	0.888	0.875	0.874	0.820	0.962
Pier	0.927	0.980	0.973	0.628	0.756	0.926	0.924
Arch	0.918	0.877	0.966	0.782	0.878	0.973	0.967
Hole	0	0	0	0	0	0	0.513
Wingwall	0	0	0	0	0	0.704	0
Average mIoU	0.894	0.904	0.942	0.762	0.836	0.856	0.841

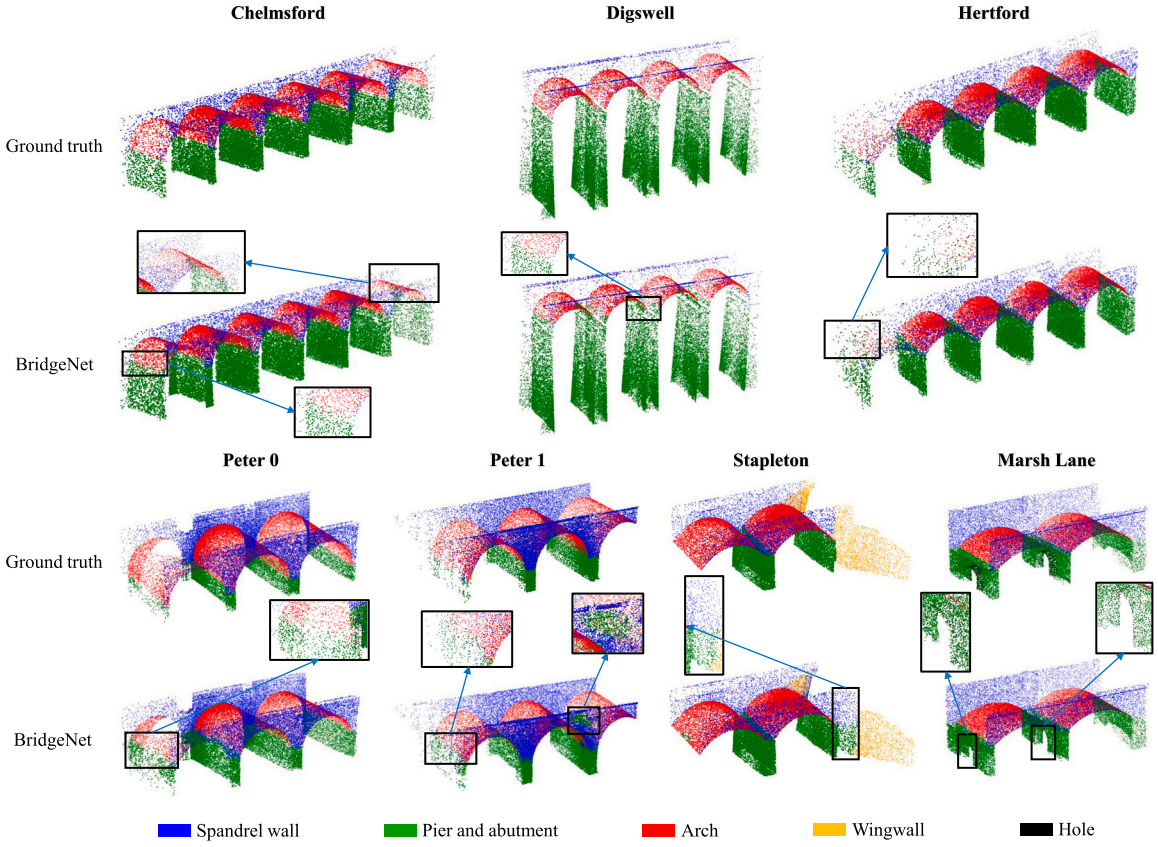


Fig. 12. Visualizations demonstrating the semantic segmentation results from BridgeNet compared with the 'ground truth' for the real bridge test dataset. Misclassifications are highlighted and shown inset with a zoomed-in view within black bounding boxes.

5. Post-processing

5.1. Instance segmentation

Since the network only gives the semantic segmentation results, the unsupervised learning method DBSCAN [57] is used to perform instance segmentation to cluster points into separate objects as shown in Fig. 13. The performance of the algorithm depends on two parameters *eps* and *min_samples*, which are the maximum distance between two samples for one to be considered in the neighbourhood of the other and the number of samples in a neighbourhood for a point to be considered as a core point, respectively. Two parameters used in the algorithm had to be adjusted following a trial and error approach to achieve the best instance segmentation performance as shown in Table 7. The outliers were automatically removed by filtering out clusters that contain points lower than the defined threshold.

5.2. Geometric feature extraction

Two difficult to obtain geometric parameters, the generatrix and the arch radius, are extracted from the manually segmented 'ground truth' point clouds as true values. These are then compared to geometric parameters extracted from segmented arch and pier point clouds by BridgeNet and DBSCAN.

The generatrix and radius of arches can be calculated using a Random Sample Consensus (RANSAC) based iterative geometric extraction algorithm. The algorithm first fits a 3D cylinder function to a random selection of a small set of points (20 points in our case) from an arch class. The shortest distance between each remaining point and the fitted generatrix (uniquely determined by the centre and the direction of the generatrix) is then calculated. By subtracting the fitted radius of the cylinder e_{radius} from this value, the residual error of the radius is obtained for each point. The number of points on the cylinder, n_{c_fit}, can then be determined by filtering points with residual errors larger than a pre-defined threshold. Then, RANSAC is applied to iteratively select points to fit the 3D cylinder and output two metrics: n_{c_fit} and e_{radius}. Those metrics are evaluated at every loop until the residual error becomes smaller than a pre-defined error threshold or the maximum iteration number is reached.

The extracted geometric parameters in the final step of the algorithm

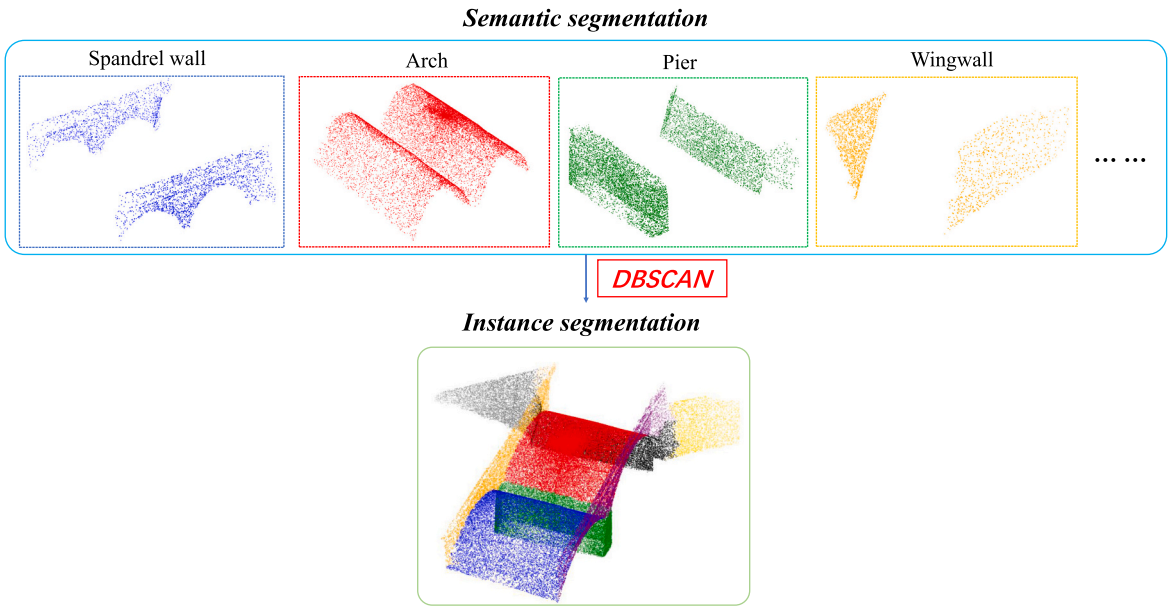


Fig. 13. Example of instance segmentation for all bridge components of the Stapleton Road viaduct by using DBSCAN.

Table 7
Two parameters of DBSCAN used for instance segmentation of each bridge after semantic segmentation.

		Chelmsford	Digswell	Hertford	Peter 0	Peter 1	Stapleton	Marsh Lane
Arch	*eps*	0.03	0.03	0.03	0.04	0.04	0.03	0.03
	min_samples	50	40	50	40	40	40	40
Pier	*eps*	0.1						
	min_samples	100						

for the segmented arches are compared to the ground truth results in Table 8 for the Chelmsford ViaductFOR. Root mean squared errors (RMSE) are specified since the geometric extraction was conducted for multiple arches. The RMSE for the radius is smaller than 1 cm, indicating remarkably accuracy. The direction of the generatrix is captured with an RMSE of less than 0.2° which shows the robustness of the segmentation results. This is demonstrated visually with a top view of the arch point clouds and their generatrix (represented by black arrows) in Fig. 14. These results demonstrate that the developed algorithm is capable of performing geometric extraction accurately even for the incomplete scans of arch 1 and arch 6.

6. Conclusions

The development of automated and generalisable techniques for semantic and instance segmentation can facilitate the assessment, inspection and monitoring of masonry railway bridges. In this work, a new deep learning network called BridgeNet was developed to perform partial segmentation on large-scale masonry point clouds. This network was inspired by RandLA-Net [50] and FG-Net [51] and modified based on their architecture to conduct semantic segmentation on large-scale point clouds. Instead of deformable convolution kernels, BridgeNet applies a 1×1 convolution layer to capture local feature correlations; this approach was observed to be more robust in the experiments. Furthermore, maxpooling is used rather than summations to aggregate neighbour point features, which alleviated overfitting issues.

Due to the scarcity of real data for training, a masonry arch bridge point cloud simulator was developed. BridgeNet was then trained with a solely synthetic dataset from this simulator and tested on a dataset composed of real point clouds from 7 railway bridges in the UK, each with significantly different geometric properties. The results were used to evaluate the performance of BridgeNet compared with two state-of-art neural networks PointNet++ and RandLA-Net. The results demonstrate that BridgeNet segments the point clouds with remarkable accuracy, and outperforms other networks by a significant margin. This result also highlights the success of the synthetic point cloud simulator in capturing global geometric properties of real bridges and justifies the research methodology developed in this paper. Segmentation errors occasionally arise due to the simplistic representation of geometry in the bridge simulator.

An unsupervised method called DBSCAN was used for instance segmentation. This enabled the investigation of key geometric parameters of the bridge. In particular, by fitting cylinders to arch instances using a RANSAC based approach, two key geometric parameters, arch radius of curvature and generatrix direction, were obtained. The geometric parameters obtained from point clouds segmented by BridgeNet, were compared to ground truth values, demonstrating a high accuracy. This demonstration emphasized the robustness of segmentation and highlighted the useful geometric information that can be gained from segmented point clouds.

Currently, our synthetic dataset simplifies the geometry of real masonry arch bridges (e.g. by excluding components, such as parapets) and the simulation of laser scanning; these aspects may be limiting the segmentation accuracy. Furthermore, in the current pipeline, BridgeNet performs semantic segmentation only, and unsupervised clustering

Table 8
RMSE is associated with the radius and the generatrix angle obtained by fitting cylinders to arch point clouds of the Chelmsford viaduct segmented from BridgeNet. The errors are relative to the radius and generatrix estimations from cylinders fitted manually sampled point clouds ('ground truth').

Arch num	Radius(m)	Generatrix(°)
6	$8.43*10^{-3}$	$1.70*10^{-1}$

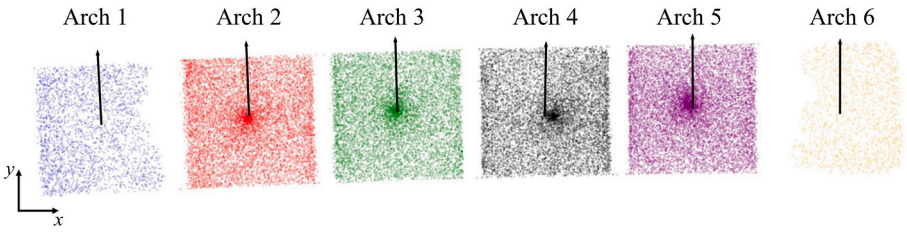


Fig. 14. Arches segmented by BridgeNet are projected on the *x-y* plane. The extracted generatrix from predicted arches is shown in black arrows.

methods, such as DBSCAN, are necessary for instance segmentation. Since hyper-parameters for DBSCAN needs to be adjusted for different cases, an end-to-end network might be proposed in the future to achieve a more efficient pipeline.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

The authors acknowledge Dan Brackenbury and Robert Gayer (University of Cambridge) and Matthew DeJong (University of Berkeley) who collected and shared the masonry arch bridge point cloud data used in this study.

References

[1] Z. Orbán, Assessment, reliability and maintenance of masonry arch railway bridges in Europe, in: P. Roca, C. Molins (Eds.), Arch Bridges IV–Advances in Assessment, Structural Design and Construction 2004, 2004, pp. 152–161. Barcelona.
[2] S. Acikgoz, M.J. DeJong, K. Soga, Sensing dynamic displacements in masonry rail bridges using 2D digital image correlation, Struct. Control. Health Monit. 25 (8) (2018), e2187, https://doi.org/10.1002/stc.2187.
[3] T. Luhmann, S. Robson, S. Kyle, J. Boehm, Close-range photogrammetry and 3D imaging, in: Close-Range Photogrammetry and 3D Imaging, de Gruyter, Berlin, Boston, 2019, https://doi.org/10.1515/9783110607253.
[4] B. Riveiro, P. Morer, P. Arias, I. De Arteaga, Terrestrial laser scanning and limit analysis of masonry arch bridges, Constr. Build. Mater. 25 (4) (2011) 1726–1735, https://doi.org/10.1016/j.conbuildmat.2010.11.094.
[5] J. Armesto, J. Roca-Pardiñas, H. Lorenzo, P. Arias, Modelling masonry arches shape using terrestrial laser scanning data and nonparametric methods, Eng. Struct. 32 (2) (2010) 607–615, https://doi.org/10.1016/j.engstruct.2009.11.007.
[6] R. Schnabel, R. Wahl, R. Klein, Efficient RANSAC for point-cloud shape detection, in: Computer Graphics Forum, Blackwell Publishing Ltd., Oxford, UK, June 2007, pp. 214–226, https://doi.org/10.1111/j.1467-8659.2007.01016.x. Vol. 26, No. 2.
[7] C. Ye, S. Acikgoz, S. Pendrigh, E. Riley, M.J. DeJong, Mapping deformations and inferring movements of masonry arch bridges using point cloud data, Eng. Struct. 173 (2018) 530–545, https://doi.org/10.1016/j.engstruct.2018.06.094.
[8] E. Valero, A. Forster, F. Bosché, E. Hyslop, L. Wilson, A. Turmel, Automated defect detection and classification in ashlar masonry walls using machine learning, Autom. Constr. 106 (2019), 102846, https://doi.org/10.1016/j.autcon.2019.102846.
[9] J. Armesto-González, B. Riveiro-Rodríguez, D. González-Aguilera, M.T. Rivas-Brea, Terrestrial laser scanning intensity data applied to damage detection for historical buildings, J. Archaeol. Sci. 37 (12) (2010) 3037–3047, https://doi.org/10.1016/j.jas.2010.06.031.
[10] D.F. Laefer, L. Truong-Hong, H. Carr, M. Singh, Crack detection limits in unit based masonry with terrestrial laser scanning, Ndt & E International 62 (2014) 66–76, https://doi.org/10.1016/j.ndteint.2013.11.001.
[11] S. Acikgoz, L. Pelecanos, G. Giardina, J. Aitken, K. Soga, Distributed sensing of a masonry vault during nearby piling, Struct. Control. Health Monit. 24 (3) (2017), e1872, https://doi.org/10.1002/stc.1872.
[12] S. Acikgoz, K. Soga, J. Woodhams, Evaluation of the response of a vaulted masonry structure to differential settlements using point cloud data and limit analyses, Constr. Build. Mater. 150 (2017) 916–931, https://doi.org/10.1016/j.conbuildmat.2017.05.075.
[13] S. Acikgoz, A. Luciano, M. Dewhirst, M.J. Dejong, R. Mair, Innovative monitoring of the response of a heritage masonry building to nearby tunnelling in London Clay, Géotechnique 72 (3) (2022) 200–215, https://doi.org/10.1680/jgeot.19.P.243.
[14] 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.
[15] R. Lu, I. Brilakis, C.R. Middleton, Detection of structural components in point clouds of existing RC bridges, Comput.-Aided Civil Infrastruct. Eng. 34 (3) (2019) 191–212, https://doi.org/10.1111/mice.12407.
[16] E. Grilli, F. Remondino, Machine learning generalisation across different 3D architectural heritage, ISPRS Int. J. Geo Inf. 9 (6) (2020) 379, https://doi.org/10.3390/ijgi9060379.
[17] S. Teruggi, E. Grilli, M. Russo, F. Fassi, F. Remondino, A hierarchical machine learning approach for multi-level and multi-resolution 3D point cloud classification, Remote Sens. 12 (16) (2020) 2598, https://doi.org/10.3390/rs12162598.
[18] R. Pierdicca, M. Paolanti, F. Matrone, M. Martini, C. Morbidoni, E.S. Malinverni, A. M. Lingua, Point cloud semantic segmentation using a deep learning framework for cultural heritage, Remote Sens. 12 (6) (2020) 1005, https://doi.org/10.3390/rs12061005.
[19] F. Matrone, E. Grilli, M. Martini, M. Paolanti, R. Pierdicca, F. Remondino, Comparing machine and deep learning methods for large 3D heritage semantic segmentation, ISPRS Int. J. Geo Inf. 9 (9) (2020) 535, https://doi.org/10.3390/ijgi9090535.
[20] 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.
[21] B. Wu, A. Wan, X. Yue, K. Keutzer, Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), IEEE, May 2018, pp. 1887–1893, https://doi.org/10.1109/ICRA.2018.8462926.
[22] B. Wu, X. Zhou, S. Zhao, X. Yue, K. Keutzer, Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud, in: 2019 International Conference on Robotics and Automation (ICRA), IEEE, May 2019, pp. 4376–4382, https://doi.org/10.1109/ICRA.2019.8793495.
[23] D. Griffiths, J. Boehm, SynthCity: a large scale synthetic point cloud, arXiv preprint (2019), https://doi.org/10.48550/arXiv.1907.04758 arXiv:1907.04758.
[24] J. Fang, D. Zhou, F. Yan, T. Zhao, F. Zhang, Y. Ma, R. Yang, Augmented lidar simulator for autonomous driving, IEEE Robot. Auto. Lett. 5 (2) (2020) 1931–1938, https://doi.org/10.1109/LRA.2020.2969927.
[25] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? The Kitti vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, June 2012, pp. 3354–3361, https://doi.org/10.1109/CVPR.2012.6248074.
[26] J.W. Ma, T. Czerniawski, F. Leite, Semantic segmentation of point clouds of building interiors with deep learning: augmenting training datasets with synthetic BIM-based point clouds, Autom. Constr. 113 (2020), 103144, https://doi.org/10.1016/j.autcon.2020.103144.
[27] C. Morbidoni, R. Pierdicca, M. Paolanti, R. Quattrini, R. Mammoli, Learning from synthetic point cloud data for historical buildings semantic segmentation, J. Comput. Cult. Herit. (JOCCH) 13 (4) (2020) 1–16, https://doi.org/10.1145/3409262.
[28] 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), 107850, https://doi.org/10.1016/j.ymssp.2021.107850.
[29] Y. Zhou, O. Tuzel, Voxelnet: End-to-end learning for point cloud based 3d object detection, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4490–4499, https://doi.org/10.1109/CVPR.2018.00472.
[30] Ö. Çiçek, A. Abdulkadir, S.S. Lienkamp, T. Brox, O. Ronneberger, 3D U-Net: learning dense volumetric segmentation from sparse annotation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, Springer, Cham, October 2016, pp. 424–432, https://doi.org/10.1007/978-3-319-46723-8_49.
[31] B. Li, 3d fully convolutional network for vehicle detection in point cloud, in: 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2017, September, pp. 1513–1518, https://doi.org/10.1109/IROS.2017.8205955.
[32] C. Choy, J. Gwak, S. Savarese, 4d spatio-temporal convnets: Minkowski convolutional neural networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 3075–3084, https://doi.org/10.1109/CVPR.2019.00319.

[33] R. Cheng, R. Razani, E. Taghavi, E. Li, B. Liu, (AF)^2-S3Net: Attentive feature fusion with adaptive feature selection for sparse semantic segmentation network, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 12547–12556, https://doi.org/10.1109/CVPR46437.2021.01236.
[34] G. Riegler, A. Osman Ulusoy, A. Geiger, Octnet: Learning deep 3d representations at high resolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3577–3586, https://doi.org/10.1109/CVPR.2017.701.
[35] Z. Liu, H. Tang, Y. Lin, S. Han, Point-voxel cnn for efficient 3d deep learning, Adv. Neural Inf. Proces. Syst. 32 (2019), https://doi.org/10.48550/arXiv.1907.03739.
[36] M. Simonovsky, N. Komodakis, Dynamic edge-conditioned filters in convolutional neural networks on graphs, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3693–3702, https://doi.org/10.1109/CVPR.2017.11.
[37] Y. Wang, Y. Sun, Z. Liu, S.E. Sarma, M.M. Bronstein, J.M. Solomon, Dynamic graph cnn for learning on point clouds, Acm Trans. Graph. (tog) 38 (5) (2019) 1–12, https://doi.org/10.1145/3326362.
[38] L. Wang, Y. Huang, Y. Hou, S. Zhang, J. Shan, Graph attention convolution for point cloud semantic segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 10296–10305, https://doi.org/10.1109/CVPR.2019.01054.
[39] L. Landrieu, M. Simonovsky, Large-scale point cloud semantic segmentation with superpoint graphs, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4558–4567, https://doi.org/10.1109/CVPR.2018.00479.
[40] L. Landrieu, M. Boussaha, Point cloud oversegmentation with graph-structured deep metric learning, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 7440–7449, https://doi.org/10.1109/CVPR.2019.00762.
[41] C.R. Qi, H. Su, K. Mo, L.J. Guibas, Pointnet: deep learning on point sets for 3d classification and segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 652–660, https://doi.org/10.1109/CVPR.2017.16.
[42] C.R. Qi, L. Yi, H. Su, L.J. Guibas, Pointnet++: deep hierarchical feature learning on point sets in a metric space, Adv. Neural Inf. Proces. Syst. 30 (2017). https://dl.acm.org/doi/10.5555/3295222.3295263.
[43] M. Jiang, Y. Wu, T. Zhao, Z. Zhao, C. Lu, Pointsift: a sift-like network module for 3d point cloud semantic segmentation, arXiv preprint (2018), https://doi.org/10.48550/arXiv.1807.00652 arXiv:1807.00652.
[44] Y. Liu, B. Fan, S. Xiang, C. Pan, Relation-shape convolutional neural network for point cloud analysis, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8895–8904, https://doi.org/10.1109/CVPR.2019.00910.
[45] H. Thomas, C.R. Qi, J.E. Deschaud, B. Marcotegui, F. Goulette, L.J. Guibas, Kpconv: Flexible and deformable convolution for point clouds, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6411–6420, https://doi.org/10.1109/ICCV.2019.00651.
[46] Y. Liu, B. Fan, G. Meng, J. Lu, S. Xiang, C. Pan, Densepoint: Learning densely contextual representation for efficient point cloud processing, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 5239–5248, https://doi.org/10.1109/ICCV.2019.00534.
[47] W. Wu, Z. Qi, L. Fuxin, Pointconv: deep convolutional networks on 3d point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9621–9630, https://doi.org/10.1109/CVPR.2019.00985.
[48] S.V. Sheshappanavar, C. Kambhamettu, A novel local geometry capture in pointnet ++ for 3d classification, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, 2020, pp. 262–263, https://doi.org/10.1109/CVPRW50498.2020.00139.
[49] Z. Zhang, B.S. Hua, S.K. Yeung, Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 1607–1616, https://doi.org/10.1109/ICCV.2019.00169.
[50] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, A. Markham, Randla-net: Efficient semantic segmentation of large-scale point clouds, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11108–11117, https://doi.org/10.1109/CVPR42600.2020.01112.
[51] K. Liu, Z. Gao, F. Lin, B.M. Chen, FG-Net: fast large-scale LiDAR point clouds understanding network leveraging correlated feature mining and geometric-aware modelling, arXiv preprint (2020), https://doi.org/10.48550/arXiv.2012.09439 arXiv:2012.09439.
[52] A.X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, F. Yu, Shapenet: an information-rich 3d model repository, arXiv preprint (2015), https://doi.org/10.48550/arXiv.1512.03012 arXiv:1512.03012.
[53] W. Wang, R. Yu, Q. Huang, U. Neumann, Sgpn: Similarity group proposal network for 3d point cloud instance segmentation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 2569–2578, https://doi.org/10.48550/arXiv.1711.08588.
[54] B. Zhang, P. Wonka, Point cloud instance segmentation using probabilistic embeddings, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 8883–8892, https://doi.org/10.1109/CVPR.2018.00272.
[55] L. Zhao, W. Tao, Jsnet: joint instance and semantic segmentation of 3d point clouds, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2020, April, pp. 12951–12958, https://doi.org/10.1609/aaai.v34i07.6994. Vol. 34, No. 07.
[56] F. Yu, K. Liu, Y. Zhang, C. Zhu, K. Xu, Partnet: a recursive part decomposition network for fine-grained and hierarchical shape segmentation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9491–9500, https://doi.org/10.1109/CVPR.2019.00972.
[57] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in: kdd, August 1996, pp. 226–231. Vol. 96, No. 34, https://dl.acm.org/doi/10.5555/3001460.3001507.
[58] A. Brencich, R. Morbiducci, Masonry arches: historical rules and modern mechanics, Int. J. Archit. Herit. 1 (2) (2007) 165–189, https://doi.org/10.1080/15583050701312926.
[59] C. Melbourne, L.D. McKibbins, N. Sawar, C.S. Gaillard, Masonry Arch Bridges: Condition Appraisal and Remedial Treatment, CIRIA, London, 2006. ISBN: 978-0-86017-656-5.