ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Volume XI-2-2026
XXV ISPRS Congress 2026 “From Imagery to Understanding”, Commission II, 4–11 July 2026, Toronto, Canada

Semantic Segmentation of Textured Non-manifold 3D Meshes using Transformers

Mohammadreza Heidarianbaei, Max Mehltretter, Franz Rottensteiner

Institute of Photogrammetry and GeoInformation, Leibniz University Hannover, Germany
(heidarianbaei, mehltretter, rottensteiner)@ipi.uni-hannover.de

Keywords: Textured Meshes, Semantic Segmentation, Transformers, Deep Learning, Cultural Heritage.

Abstract

Textured 3D meshes jointly represent geometry, topology, and appearance, yet their irregular structure poses significant challenges for deep-learning-based semantic segmentation. While a few recent methods operate directly on meshes without imposing geometric constraints, they typically overlook the rich textural information also provided by such meshes. We introduce a texture-aware transformer that learns directly from raw pixels associated with each mesh face, coupled with a new hierarchical learning scheme for multi-scale feature aggregation. A texture branch summarizes all face-level pixels into a learnable token, which is fused with geometrical descriptors and processed by a stack of Two-Stage Transformer Blocks (TSTB), which allow for both a local and a global information flow. We evaluate our model on the Semantic Urban Meshes (SUM) benchmark and a newly curated cultural-heritage dataset comprising textured roof tiles with triangle-level annotations for damage types. Our method achieves 81.9% mF1 and 94.3% OA on SUM and 49.7% mF1 and 72.8% OA on the new dataset, substantially outperforming existing approaches.

1. Introduction

Understanding 3D structure is a fundamental task in various domains, e.g. Geospatial Analysis (Kölle et al., 2021) and Cultural Heritage (CH) preservation (Yang et al., 2023). Textured 3D meshes offer unique advantages over alternative 3D surface representations such as point clouds or voxel grids: Such meshes represent the geometry and topology of 3D objects by connecting vertices by edges and faces. Additionally, they contain textural information, with every face being linked to a corresponding area in a 2D image (texture map), enabling the integration of surface appearance attributes such as colour, pattern, and material properties at a higher spatial resolution than the point cloud corresponding to its vertices.

However, due to the irregular structure of 3D textured meshes, their semantic segmentation remains a challenge for deep learning methods. Existing methods for processing meshes (Feng et al., 2019; Hu et al., 2022; Milano et al., 2020) still exhibit limitations. A major problem is the reliance on topological constraints (e.g. manifoldness) that are often violated in real-world meshes, which restricts the applicability of such methods. While some works attempt to handle non-manifold meshes, e.g. (Sharp et al., 2022; Heidarianbaei et al., 2025), most of them focus on geometrical features, considering texture in simplistic ways only. Some methods rely on low-dimensional descriptors, e.g. a single RGB vector for every vertex (Zi et al., 2024). Such a representation is spatially sparse and fails to capture high-frequency details available in the texture map. Other approaches resort to statistical descriptors to represent texture, e.g. mean colour vectors or histograms. While such descriptors provide a compact representation, they disregard spatial patterns in the texture that may be crucial for distinguishing different object types. This difficulty is partly due to the fact that the number of texture pixels associated with each face can vary significantly, making the direct application of conventional architectures, e.g., convolutional neural networks, non-trivial.

Recent work on semantic segmentation has increasingly relied on transformer-based architectures based on the attention mechanism (Vaswani et al., 2017) due to the ability of such networks to consider global context. However, their computational complexity often renders full attention impractical, leading to the requirement for patchification, i.e. a reduction of the spatial resolution of the input data. In the domain of 3D mesh processing, Heidarianbaei et al. (2025) presented a transformer-based approach that avoids a reduction of the spatial resolution, addressing this problem by partitioning mesh faces into clusters and applying self-attention within each cluster to capture local context while also determining one token encoding the entire cluster (*cluster token*). The cluster tokens interact with each other and with all face tokens via cross-attention to model global dependencies. We believe that, despite the residual connections used in transformers, this formulation might lead to an over-smoothing of the classification results, as face-level tokens are updated by weighted averages of cluster tokens.

In this paper, we propose a new method for semantic segmentation of textured 3D meshes. It builds on (Heidarianbaei et al., 2025) due to its capacity for processing non-manifold meshes, but we extend it to address the limitations of existing methods mentioned earlier. On the one hand, we introduce a transformer-based texture branch that directly processes all pixel values associated with each face and delivers a texture feature vector that is fused with geometrical features, resulting in a unique representation of every face of the mesh by a vector capturing both cues. On the other hand, we propose a modified hierarchical attention mechanism that enhances inter-cluster information exchange while preserving local feature diversity and, thus, reducing potential oversmoothing effects of the original cross-attention-based approach. This is achieved by restricting the global information exchange to cluster tokens via self-attention, after which each cluster token propagates the aggregated context back to the faces of its cluster. This process enables an efficient bidirectional information flow between local and global levels, considering context at different scales without blurring fine-grained local features.

We evaluate our proposed method using two datasets, the SUM benchmark dataset for urban classification (Gao et al., 2021) and a new dataset from the domain of Cultural Heritage pre-

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XI-2-2026-463-2026

servation consisting of textured 3D mesh patches of the roof of a historical building with triangle-level annotations for damages such as *biological colonization*, showing the applicability of the method to different domains. Our main contributions can be summarized as follows:

- We introduce a new unified transformer-based architecture for semantic segmentation of textured 3D meshes that jointly considers geometrical and textural features, deriving the latter directly from the pixels associated with each mesh face. Unlike prior approaches relying on coarse colour statistics or vertex-level features, our method explicitly aggregates per-face texture information through a dedicated texture branch.
- We propose a new method for capturing long-range interactions in transformer-based semantic segmentation of 3D meshes to facilitate effective information exchange between local and global contexts while avoiding over-smoothing and preserving fine-grained geometric detail.
- We evaluate our approach on the two datasets mentioned earlier, showing its applicability in two different application domains (urban classification, cultural heritage preservation). We also report the results of ablation studies validating the effectiveness of our design choices.

2. Related Work

Processing 3D data has witnessed significant advancements in recent years, largely driven by progress in deep learning (Zhang et al., 2025; Ioannidou et al., 2017). Most existing methods focus on a specific representation, e.g. on voxel-based representations of 3D data (Maturana and Scherer, 2015; Wu et al., 2015; Riegler et al., 2017) or on point clouds (Qi et al., 2017; Hu et al., 2020; Qian et al., 2022), which provide a more efficient and explicit way to represent the geometry of 3D objects. Despite these advancements, such methods cannot directly operate on 3D meshes as input. Meshes are based on an inherently irregular data structure, posing significant challenges for neural network architectures originally designed for regular grids. As a result, mesh-based semantic segmentation is still underexplored. A common workaround is to convert mesh faces into point clouds, e.g. using face centroids with texture descriptors (Laupheimer, 2022), and then to apply networks for point cloud segmentation, e.g. (Qi et al., 2017). However, it would be desirable to fully exploit the inherent potential of mesh data.

There are just a few works on the direct classification of triangles in a mesh. Early work extended convolutional neural networks to operate directly on 3D meshes by redefining convolution and pooling for non-Euclidean geometry. For instance, geodesic CNNs (Masci et al., 2015) apply convolution to curved surfaces using geodesic coordinates instead of grid structures. MeshNet (Feng et al., 2019) processes mesh faces using spatial and structural descriptors and expands the receptive field by mesh-based convolution layers. Hu et al. (2022) focus on hierarchical feature learning, including subdivision-based pooling to build fine-to-coarse mesh representations. Hanocka et al. (2019) consider edges to be the basic elements and use an edge-based pooling strategy to adapt mesh topology and capture multi-scale structure. Another research direction leverages the graph structure of meshes. Milano et al. (2020) extend pointwise convolutions to meshes by constructing face- and edge-based graphs and aggregating features via graph attention networks, using mesh simplification for pooling. MeshWalker (Lahav and Tal, 2020) performs random walks on the vertex graph and encodes the resulting sequences with recurrent networks to capture local topology. However, all of these methods require manifold mesh structures, enforcing well-defined neighbourhood relationships that enable the adaptation of conventional deep learning operations. In practice, 3D meshes produced in automated photogrammetric pipelines often violate these manifold constraints, limiting the applicability of such approaches to real-world data (Kölle et al., 2021; Gao et al., 2021).

There is some work that relaxes manifold assumptions to better handle irregular or non-manifold geometry. Laplacian2Mesh (Dong et al., 2022) operates in the spectral domain by leveraging the mesh Laplacian, enabling processing of non-manifold structures but focusing primarily on vertex-level tasks and risking loss of fine spatial detail. DiffusionNet (Sharp et al., 2022) performs directional filtering using pointwise perceptrons with learned diffusion and spatial gradients, though aggregation remains limited to local neighbourhoods. In contrast, Tutzauer et al. (2019) extract multi-scale face descriptors and apply a 1D CNN, with spherical neighbourhoods providing geometric context; the convolution is mainly used for feature embedding and not for true spatial aggregation. None of the methods cited so far incorporates textural information, and their focus remains primarily on geometric and object-level data.

MFSM-Net (Hao et al., 2025) represents texture by a single orthophoto derived from the mesh. Textural and geometrical features are extracted from the orthophoto and the mesh in two separate streams before being fused. However, a single image cannot fully represent the radiometric information of all mesh faces in real 3D scenarios. More recently and most relevant to our method, Heidarianbaei et al. (2025) introduced a transformer-based architecture for the per-face semantic segmentation of textured non-manifold meshes. They construct compact face descriptors by combining handcrafted geometrical features with simple texture statistics. To cope with the irregular mesh structure and the large number of faces, the faces are spatially clustered. A Local–Global (L-G) transformer architecture captures both local and global context: local self-attention operates within clusters with the support of cluster tokens, while a global cross-attention mechanism enables information flow across clusters at substantially reduced computational cost. However, that method does not directly process the raw texture information associated with each face, but relies on few statistical descriptors that are not expected to capture fine-grained textural patterns. Second, the cross-attention mechanism in the L-G blocks updates the per-face features as weighted averages of cluster tokens, which we believe to lead to oversmoothing of the classification results.

To address the limitations of the cited approaches, we propose a new network architecture for semantic segmentation of 3D meshes. The new method extends (Heidarianbaei et al., 2025) by a texture-embedding module and an improved transformer-based mechanism for considering global context. To the best of our knowledge, no existing deep learning framework directly integrates raw texture information alongside geometrical features at triangle level for the task at hand; the exception (Hao et al., 2025) requires a projection to a common plane, which might be problematic in real 3D scenarios. We also believe that the proposed attention-based mechanism will allow the model to propagate high-level contextual knowledge back to individual faces without overriding the local information completely, thus allowing to predict class labels correctly at a fine local scale.

This contribution has been peer-reviewed. The double-blind peer-review was conducted on the basis of the full paper.
https://doi.org/10.5194/isprs-annals-XI-2-2026-463-2026 | © Author(s) 2026. CC BY 4.0 License.

3. Methodology

3.1 Overview

The input to our method consists of a textured 3D triangulated mesh $M = (V, F, T, I)$, where V denotes the set of vertices, each corresponding to a point in 3D space, and F denotes the set of triangular faces. Each face is defined by an ordered list of three vertices. The texture is contained in a texture image I; T denotes the information required for linking every face to an area in I, i.e. the coordinates of the points in I corresponding to the three vertices of the face (*texture coordinates*). The primary objective of the network is to perform semantic segmentation. The output is represented by $\widehat{Y} \in \mathbb{R}^{|F| \times C}$, where $|F|$ is the number of faces and C is the number of predefined classes. $\widehat{Y}$ contains a vector of C class scores for every face of the mesh. The class label of a face is determined as the class having the maximum score.

As shown in Figure 1, the input mesh is first processed by a feature extraction branch, which generates a feature vector F_{N_F} of dimension N_F for every triangular face in the mesh. This vector encodes both geometrical and textural information related to a face. However, unlike most existing work, our network directly learns to determine representations from the raw pixel data using a transformer-based network. The resulting per-face feature vectors are aggregated into a tensor of size $|F| \times N_F$, which is then passed to the subsequent components of the network. The feature extraction branch is described in Section 3.2.

The core module of our network builds upon transformer architectures (Vaswani et al., 2017). However, the computational cost of standard self-attention scales quadratically with the number of faces $|F|$, which becomes prohibitive for meshes containing a large number of faces. To overcome this limitation, inspired by prior works (Chu et al., 2021; Heidarianbaei et al., 2025), we adopt a hybrid local–global processing strategy and propose the so-called Two-Stage Transformer Block (TSTB), in which face feature vectors are processed in two successive stages operating on a local and a global level, respectively.

Following (Heidarianbaei et al., 2025), we first construct a spatial partition of the mesh: the vertices V are clustered into K subsets using K-means clustering (Bishop, 2006). Each face is assigned to the cluster that contains the majority of its vertices; if all three vertices lie in different clusters, the face will be assigned randomly. The input feature tensor is reshaped so that there is one sequence of feature vectors (*tokens*) per cluster, and a single learnable cluster token is prepended to each sequence. This results in a tensor $\mathbf{X}$ of dimension $K \times (S_c + 1) \times N_F$, where S_c denotes the maximum number of faces contained in any cluster. To enable efficient implementation and parallel processing across clusters, all clusters with fewer than S_c faces are zero-padded to this size; a binary attention mask is generated to identify the padded feature vectors, so that attention will only be computed among valid (non-padded) tokens. In the first stage of a TSTB, self-attention is restricted to operate within the clusters. This localized attention reduces the computational footprint while preserving fine-scale cues. To enable long-range information exchange, we propose to use a self-attention mechanism among cluster tokens in the second stage of a TSTB, which enriches them with long-range contextual information. The enriched cluster tokens will interact with their local face tokens in the first stage of the following block, enhancing global–local consistency while preserving the discriminative details of individual face vectors.

The last TSTB delivers the set of face representations that serves as the input of the classification head, which discards cluster tokens and predicts the class scores and labels. The TSTB and the classification are described in Sections 3.3 and 3.4, respectively. Section 3.5 presents the training procedure.

3.2 Feature Extraction Branch

This branch extracts a feature vector for each face by combining geometrical and textural cues. It consists of two independent sub-branches: the *geometry branch* producing geometrical features $\mathbf{F}_G \in \mathbb{R}^d$, and the *texture branch* generating textural features $\mathbf{F}_T \in \mathbb{R}^d$. The two vectors are concatenated to form $\mathbf{F} = [\mathbf{F}_G, \mathbf{F}_T] \in \mathbb{R}^{2d}$, which is processed by a two-layer multilayer perceptron (MLP) to obtain the final face-specific feature vector $\mathbf{F}_{\text{out}} \in \mathbb{R}^{N_F}$. The MLP uses ReLU activations and dropout for regularization. The sub-branches are described below.

3.2.1 Geometry Branch: Similarly to (Heidarianbaei et al., 2025), we derive 16 hand-crafted features per face that encode shape and positional cues: normalized vertex coordinates, normal vector components, area, and three angles with the adjacent faces. A linear layer projects the raw descriptor $\mathbf{F}_G^{\text{raw}} \in \mathbb{R}^{16}$ into the final geometrical embedding $\mathbf{F}_G \in \mathbb{R}^d$.

3.2.2 Texture Branch: In order to determine the texture feature vectors, the pixels associated with a face are extracted from the texture image I inside the area indicated by the texture coordinates and stored in a tensor $\mathbf{F}_T^0$ of dimension $P \times Ch$, where P is the number of pixels in the face (which varies from face to face) and Ch is the number of channels ($Ch = 3$ for RGB texture images). As the model requires the same feature dimensionality for each face, the texture branch determines a texture feature vector $\mathbf{F}_T \in \mathbb{R}^d$ from this information that encodes the texture information of the face.

The texture branch, which is a variant of a transformer network (Vaswani et al., 2017), takes $\mathbf{F}_T^0$ as input. First, a learnable token (referred to as the *texture token*) is appended to the tensor $\mathbf{F}_T^0$, resulting in an augmented input tensor $\mathbf{F}_T^{\text{raw}}$ of size $(P + 1) \times Ch$. $\mathbf{F}_T^{\text{raw}}$ is processed by a fully connected layer that maps the features into a space of dimension d, resulting in a tensor $\mathbf{F}_T^{\text{inp}}$ of dimension $(P+1) \times d$ containing P transformed feature vectors and the transformed texture token. This tensor is processed by a transformer block (Vaswani et al., 2017) using multi-head self-attention. The MLP inside the block consists of two fully connected layers with ReLU activation and uses an intermediate feature dimension of $2d$. As the texture token interacts with all pixels of a face, we only retain this token after applying the transformer block and consider it to be the final texture feature $\mathbf{F}_T \in \mathbb{R}^d$ of the corresponding face.

3.3 Two-Stage Transformer Blocks (TSTB)

The TSTB is the main building block of our architecture. An overview of the structure of such a block is given in Figure 2. Each block j receives an input in the form of a tensor $\mathbf{E}_j$ of dimension $K \times (S_c + 1) \times N_F$. This tensor contains the face-specific feature vectors together with the cluster tokens. The TSTB generates an output tensor $\mathbf{Z}_j$ of the same dimension. In each TSTB, the feature vectors are updated as described below. After each block, the binary attention mask introduced in Section 3.1 is applied to reset the padded feature values to zero. The input to the first block ($j = 1$) is the tensor $\mathbf{X}$ that is generated from the outputs of the feature extraction branch and the learnable cluster tokens (Section 3.1), i.e., $\mathbf{E}_1 = \mathbf{X}$.

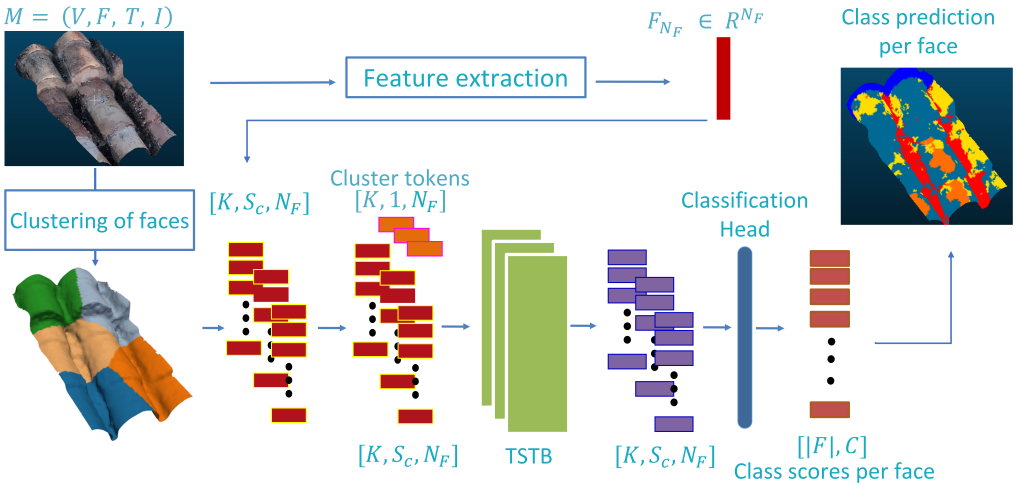

Figure 1. Overview of the network architecture. The feature extraction branch generates one feature vector per face. The faces are clustered using K-means clustering. The tensor containing the feature vectors is reshaped so that there is one sequence of feature vectors per cluster, and a learnable cluster token is prepended to each sequence. These feature vectors are processed by a sequence of N_{Bl} TSTB. The output of the final block is used to predict the class scores and labels. Numbers in brackets denote tensor dimensions.

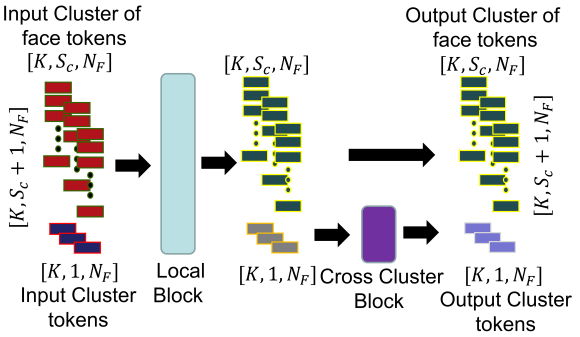

Figure 2. Architecture of a TSTB. It consists of two sub-blocks: a local block in which the feature vectors within each face cluster can interact with each other, and a cross-cluster block designed to capture global context by allowing the cluster tokens of different clusters to interact. Numbers in brackets are dimensions of tensors.

The structure of the TSTB is designed to capture both local and global context from the input mesh. As discussed in Section 3.1, applying attention across all feature vectors simultaneously would be computationally prohibitive. Therefore, each TSTB contains two transformer-based sub-blocks:

1. The **Local Block** processes the feature vectors within each cluster independently, enabling local context aggregation. This sub-block is described in Section 3.3.1.
2. The **Cross-Cluster Block** allows the cluster tokens to interact across clusters, capturing global dependencies. It is presented Section 3.3.2.

Combining local and cross-cluster blocks provides a hierarchical mechanism for progressively integrating fine-grained geometrical details with global structural context. The local block focuses on intra-cluster interactions, enabling each face feature to exchange information with its neighbours and the corresponding cluster token. On the other hand, the cross-cluster block operates on the set of cluster tokens, facilitating inter-cluster communication through self-attention between cluster tokens. By updating cluster tokens with information from all other clusters, the cross-cluster block considers long-range dependencies and ensures global consistency across the mesh. Global context flows back to individual faces when the updated cluster tokens are reintegrated into the local blocks of subsequent layers. This alternation between local and global attention enables the model to capture both detailed local relations and long-range structural patterns, which we expect to lead to more expressive representations.

3.3.1 Local Block: Each local block is structured as a standard Transformer block (Vaswani et al., 2017), denoted by TB. The input features are linearly projected into query, key, and value matrices, and multi-head self-attention is applied within each cluster of tokens. Following the attention operation, the output is normalized and passed through a two-layer MLP with ReLU activations, with residual connections applied to both sub-modules. To formalize this process, let the input tensor of block j be $\mathbf{E}_j$. For each cluster k, the corresponding input slice is defined as

$$\mathbf{E}_{k,j} = \left[\mathbf{F}_{N_F}^{Cl_k}, \mathbf{F}_{N_F}^{k_1}, \ldots, \mathbf{F}_{N_F}^{N_k}, \ldots, \mathbf{F}_{N_F}^{S_c}\right]_j \in \mathbb{R}^{(S_c+1)\times N_F},$$

where the first vector, $\mathbf{F}_{N_F}^{Cl_k}$, corresponds to the cluster token and $\mathbf{F}_{N_F}^{k_n}$ is the feature vector of the n^{th} face in cluster k; the feature vectors with a face index larger than the number N_k of faces in that cluster are zero-padded (Section 3.1). The transformer block operates on the slice according to

$$\mathbf{Z}_{k,j} = TB(\mathbf{E}_{k,j}). \quad (1)$$

After applying TB, the zero-padded feature vectors are reset to zero according to the binary attention masks. The output of the local block, $\mathbf{Z}_j^{loc}$, which collects the cluster-specific slices $\mathbf{Z}_{k,j}$, preserves the input dimensionality, i.e. its dimensions are $K\times(S_c+1)\times N_F$, but it encodes enriched local representations.

This mechanism captures intra-cluster dependencies and aggregates local geometric context among faces. Each local block independently processes the feature sequences within a cluster, enabling every feature vector to attend to and influence others within its local region. The attention-based interaction also applies to the cluster token. On one hand, each face feature vec-

tor is influenced by the cluster token of its cluster; on the other hand, the cluster token aggregates information from all face features in the cluster. This bidirectional exchange serves two key purposes: (1) the cluster token captures a summary representation of its cluster, and (2) this aggregated information is redistributed to all member features. As described in Section 3.3.2, in the cross-cluster block, cluster tokens from all clusters interact, so that in subsequent local blocks, the cluster tokens also encode global context that influences all face-level features.

3.3.2 Cross-Cluster Block: This block addresses dependencies between clusters and thus captures global context with minimal computational overhead. Only the cluster tokens participate in this process. First, the cluster tokens are extracted from the output of the local block, $\mathbf{Z}_j^{loc}$, and combined to a tensor $\mathbf{Z}_j^{cl,in}$ of dimensionality $K \times N_F$, which serves as the input to another transformer block of the same structure as the one described in Section 3.3.1. This TB generates an output tensor of the same dimension as the input:

$$\mathbf{Z}_j^{cl,out} = TB(\mathbf{Z}_j^{cl,in}). \tag{2}$$

The transformed cluster tokens are then copied back into the output tensor of the local block $\mathbf{Z}_j^{loc}$, producing the final output $\mathbf{Z}_j$ of TSTB j. This output forms the input for the next TSTB, i.e. $\mathbf{E}_{j+1} = \mathbf{Z}_j$. Thus, the updated cluster tokens propagate global information into the subsequent block. The output $\mathbf{Z}_{N_{Bl}}$ of the final TSTB block serves as the input to the classification head described in Section 3.4.

3.4 Classification Head

The classification head uses the output $\mathbf{Z}_{N_{Bl}}$ of the last TSTB to predict one class label for every face of the input mesh based on the feature vector representing that face, which has accumulated local and global information. Before doing so, the cluster tokens, the zero-padded dummy face tokens and the arrangement of the faces according to the clusters, still maintained in the structure of $\mathbf{Z}_{N_{Bl}}$, are discarded. Thus, a new tensor $\mathbf{Z}_F$ of dimension $|F| \times N_F$ is generated. Then, each of these N_F-dimensional feature vectors is transformed to a vector $\mathbf{y}_i$ containing C raw class scores using a fully connected layer. Finally, these raw class scores are normalized using the softmax function, yielding normalized class scores for face i,

$$\hat{\mathbf{y}}_i = \mathrm{softmax}(\mathbf{y}_i)\,, \tag{3}$$

collected in the tensor $\widehat{Y}$ introduced in Section 3.1. The final prediction for each face is obtained as the class label having the maximum class score.

3.5 Training

The network is trained by minimizing a categorical cross-entropy loss L_{ce}, which measures the discrepancy between the predicted class probabilities for each mesh face and the reference labels:

$$L_{ce} = -\frac{1}{M} \sum_{i \in \mathcal{B}} \sum_{c=1}^{C} y_{ic} \log(\hat{y}_{ic}). \tag{4}$$

In every iteration, Equation 4 forms the basis of an update step, considering a minibatch of the training samples. The index set of the sampled faces in the current minibatch is denoted by $\mathcal{B} \subseteq \{1, \ldots, |F|\}$, with $|\mathcal{B}| = M$. M can differ from the number of faces $|F|$, because some faces may be unlabelled. The binary indicator y_{ic} specifies whether face i belongs to class c ($y_{ic} = 1$) or not ($y_{ic} = 0$), and $\hat{y}_{ic}$ denotes the predicted probability for face i to be assigned to class c. Details about the training procedure used in the experiments can be found in Section 4.2.

4. Experiments and Results

4.1 Datasets

Two datasets were used for evaluating our method: SUM, a benchmark for urban semantic segmentation, and a new dataset designed for detecting damages in a cultural heritage site (CH).

4.1.1 Semantic Urban Meshes (SUM): This dataset, covering an urban area of approximately 4 km^2 in the centre of Helsinki (Finland), was introduced by Gao et al. (2021). It consists of textured 3D meshes reconstructed from airborne oblique imagery. The average side length of a triangle is approximately 0.8 m, and the ground sampling distance (GSD) of the texture is 10 cm. The faces are annotated with one of six semantic categories typical for urban classification (*terrain* - *T*, *high vegetation* - *V*, *building* - *Bld*, *water* - *W*, *car* - *C*, and *boat* - *B*), resulting in approximately 19 million labelled faces. The data is split into 64 tiles of about 250 m^2 each, with 40 / 12 / 12 of them to be used for training, validation and testing, respectively.

4.1.2 Damage Detection in Cultural Heritage (CH): This dataset was designed for evaluating methods for damage detection in CH assets. It was generated in the context of the project ChemiNova, in which the development of the proposed method is embedded. It is based on a textured 3D mesh of a part of the roof of the La Nau Centre in Valencia (Spain). The mesh was generated from 428 aerial RGB images acquired by the built-in camera of a DJI Mini 4 Pro drone with a focal length of 7 mm. The distance from the object was about 5 m, and the images were acquired with 80% forward and 70% side laps. Agisoft Metashape[1] was used to orient the images using Structure-from-Motion and to generate a dense 3D point cloud as well as the textured 3D mesh. The positions of the drone determined by the built-in GNSS sensor were used for georeferencing the block in a global reference frame (WGS84 / UTM 30N). The average side length of a triangle in the mesh is about 7 mm, the GSD of the texture is 1 mm.

To generate a reference, expert conservators defined a class structure of damages occurring on the roof that consists of seven categories. Two of them encode different states of undamaged areas: *No Damage* (*ND*) and *Joint* (*J*), corresponding to bright and dark undamaged areas, respectively. The remaining five classes encode actual damages: *Stain* (*St*), *Deposit* (*D*), *Loss of Structural Integrity* (*LI*), *Biological Colonization* (*BC*) and *Salt Efflorescence* (*SE*). A part of the 3D mesh was annotated by the conservation experts by digitizing polygons in an orthophoto of the roof and assigning them to the classes just mentioned; the results were projected onto the mesh to obtain precise triangle-wise annotations. Triangles without annotations were marked as being *unlabelled*. Finally, the annotated mesh was spatially partitioned into 230 tiles containing about 30k–50k triangles each, which were divided into training, validation, and test subsets. Table 1 presents the class distribution of the triangles and the assignment of tiles to the training, validation and test datasets. The class distribution is very imbalanced, with the two classes encoding undamaged areas corresponding to about 70%

[1] https://www.agisoft.com/ (accessed 13/02/2026)

of the data. The dominating damage type is *Biological Colonization* (18% of the samples), whereas the damage types *Deposit* and *Salt Efflorescence* are extremely underrepresented. In total, the CH dataset contains about 5 million annotated triangles.

Class	*Training*	*Validation*	*Testing*	*Total*
ND	1,445,547	348,638	530,823	2,325,008
J	778,955	182,090	234,293	1,195,338
St	337,499	48,771	119,975	506,245
D	8,185	1,342	2,837	12,364
LI	67,742	19,573	22,448	109,763
BC	575,191	144,778	202,052	922,021
SE	15,107	6,233	5,265	26,605
Total	3,228,226	751,425	1,117,693	5,097,344
Tiles	133	29	41	203

Table 1. Number of triangles per class and number of tiles in the CH dataset and the training, validation and test subsets.

4.2 Experimental Setup

4.2.1 Hyperparameters, Training: All experiments were conducted on a single NVIDIA A100 GPU (40 GB memory). Network hyperparameters were determined empirically based on validation set performance. The same hyperparameter values were used for both datasets, except for ablation studies. The dimensionality of the geometrical and texture feature vectors (Section 3.2) was set to $d = 64$. While transformers can, in principle, process sequences of arbitrary length, a uniform input length is required in practice. Therefore, we fix the number of pixels per face to $P = 128$ in the texture branch (Section 3.2.2). Faces containing more than 128 pixels are truncated (this occurs for fewer than 2% of samples in both datasets), whereas smaller faces are zero-padded to this length. Similarly to what is done in the TSTB (Section 3.1), an attention mask is used to ignore padded tokens during processing. Beyond the feature extraction branch, the network comprises $N_{Bl} = 6$ TSTBs, each with two attention heads and an embedding dimension of $N_F = 256$. The number of clusters (Section 3.1) was set to $K = 300$.

The network was trained on tiles from the training subsets of the datasets by minimizing the objective defined in Section 3.5. Model weights were initialized randomly according to (Glorot and Bengio, 2010). We used the AdamW optimizer (Loshchilov and Hutter, 2017) with an initial learning rate of $\eta = 1 \times 10^{-4}$. A cosine annealing schedule was used to progressively decay the learning rate to 1×10^{-6} over the first half of training epochs, following the strategy of Loshchilov and Hutter (2017). To mitigate overfitting, we applied extensive data augmentation, including random rotations of up to $\pm 45^\circ$ about all three axes, random scaling with factors in the range $[0.5, 2]$, and additive Gaussian noise ($\sigma = 0.01$) applied to vertex positions to improve robustness to geometric perturbations. The final model weights are selected based on the highest $mF1$ score achieved on the validation set during training.

4.2.2 Evaluation Metrics: Performance was evaluated by comparing the labels predicted for the test sets to the provided reference. We determined the class-specific $F1$ scores and the mean F1 score over all classes ($mF1$). For most experiments we only report $mF1$ and the overall accuracy (OA), the latter corresponding to the percentage of correctly classified faces.

4.2.3 Ablations: We also performed several ablation studies. First, we assess the impact of the embedding dimensionality N_F on the result (Section 3.2). Second, we analyze the results for different values of the number N_{Bl} of TSTBs in our model (Section 3.1). Third, we analyze the contribution of the input modalities on the classification results. This is achieved by training two additional classifiers that either only use the geometrical features $\mathbf{F}_G$ or only the texture features $\mathbf{F}_T$. These first three ablations were only performed on the SUM dataset. A fourth ablation is related to the impact of the way in which the features are processed in the TSTBs. In particular, we compare our method to another one that uses the cross-attention block of Heidarianbaei et al. (2025) instead of our cross-cluster block in the TSTB. This comparison, which is supposed to highlight the advantages of the new architecture of that block, is performed on both test datasets.

4.2.4 Baseline Methods: As the manifoldness constraint is not satisfied in either of our datasets, we had to select baselines capable of handling non-manifold meshes. Among the few existing methods (Section 2), we selected DiffusionNet (Sharp et al., 2022) and NoMeFormer (Heidarianbaei et al., 2025) as baselines, in both cases relying on the original hyperparameters reported in the respective papers. It is worth noting that DiffusionNet completely disregards textural information, while NoMeFormer relies solely on pixel statistics. The comparison to these baselines is based on the performance metrics presented in Section 4.2.2. In addition, for the SUM benchmark, we compare our results to those of the best-performing methods on that dataset according to (Gao et al., 2021): RF-MRF (Rouhani et al., 2017) and KPConv (Thomas et al., 2019). For that comparison, we had to adapt the quality metrics: Gao et al. (2021) weight each triangle by its area when computing OA and $mF1$, so we present these scores according to the same definition. KPConv is a point-cloud based method, which was applied to classify points sampled from the triangulated mesh. In this case, it is not exactly clear how the quality indices were determined; we use the numbers presented in (Gao et al., 2021).

4.3 Results

4.3.1 Results on the SUM Dataset: Table 2 summarizes the results of the evaluation of our method on the SUM dataset. The OA of 94.3% is rather satisfactory, indicating a good performance of our method: less than 6% of the class labels are wrong. The $mF1$ score is somewhat lower at 81.9%, which still seems to be acceptable. In particular, for the classes *terrain*, *high vegetation* and *building*, the F1 scores are larger than 92%, which we consider very good. However, the difference between $mF1$ and OA indicates some problems with less frequent classes. For *water*, *car* and *boat*, the F1 scores drop to 78.8%, 62.9%, and 64.3%, respectively. In particular, *car* and *boat* exhibit a high intra-class variability. With only a handful of training samples available for these classes, achieving strong generalization is inherently challenging.

Class	*T*	*V*	*Bld*	*W*	*C*	*B*
$F1$ [%]	92.3	94.5	95.6	78.8	62.9	64.2

Table 2. Class-specific F1 scores of our method achieved on the SUM dataset. The $mF1$ score is 81.9%, the OA is 94.3%.

4.3.2 Results on the CH Dataset: Table 3 presents the evaluation of the results of our method achieved on the CH dataset. The OA of 72.8% seems to be acceptable, though it is certainly lower than the one achieved on the SUM dataset. At 49.7% the $mF1$ score is considerably lower, with the difference to OA again indicating problems with underrepresented classes. Notably, the two classes corresponding to undamaged triangles

(ND and J), which together account for approximately 70% of the dataset, achieve F1 scores of about 80%, indicating a relatively strong performance in the dominant categories. In contrast, the classes D and SE have the lowest F1 scores (0.0% and 11.7%, respectively). This performance drop is positively correlated with the number of samples in these categories, highlighting the severe class imbalance and underrepresentation that limit the model's ability to learn minority class features. The damage classes with the highest F1 scores are LI and BC. The latter represents the type of damage occurring most frequently in the data. The relatively strong performance for LI, for which the number of samples is not very large, is more surprising; perhaps this damage type leads to specific geometrical features. These results indicate the potential of the method for such challenging datasets, but also highlight its limitations.

Class	ND	J	St	D	LI	BC	SE
$F1$ [%]	78.3	81.8	50.1	0.0	62.9	63.4	11.7

Table 3. Class-specific F1 scores of our method achieved on the CH dataset. The $mF1$ score is 49.7%, the OA is 72.8%.

4.3.3 Ablation Studies: Table 4 shows the $mF1$ scores and overall accuracies achieved on the SUM test set when using our method with different values of the feature dimension N_F. Increasing N_F from 32 to 256 improved the $mF1$ scores and OA values by 7% and 3.5%, respectively, which shows that this hyperparameter has a significant impact on the results. The table indicates that using larger values might still lead to an increase in the performance, but this could not be tested due to hardware constraints.

N_F	OA [%]	$mF1$ [%]
32	90.6	74.9
64	90.8	75.4
128	92.9	78.6
256	94.3	**81.9**

Table 4. Effect of the feature dimension N_F on model performance on the SUM dataset.

A comparison of the results achieved on the SUM test set by using different numbers N_{Bl} of TSTBs showed that using only two such blocks resulted in a suboptimal performance of 78.5% in $mF1$, which compares to a $mF1$ score of 81.9% achieved with $N_{Bl} = 6$ blocks, our default configuration. Further increasing the number of blocks to $N_{Bl} = 8$ resulted in an insignificant improvement (smaller than 0.05%).

Table 5 reports the performance metrics achieved by our method when using different input modalities. Using geometry alone yields the lowest $mF1$ score and OA of 66.9% and 84.1%, respectively, indicating that geometric cues are insufficient for reliable classification on the SUM dataset. Using texture alone results in an improvement of $mF1$ by 3.3% ($mF1 = 70.2\%$). This suggests that texture carries more discriminative information. However, combining both geometry and texture achieves a substantially higher $mF1$ score and OA of 81.9% and 94.3%, respectively, demonstrating the complementary nature of the two modalities and the advantages of their joint usage.

Finally, we compare two different strategies for modeling long-range interactions. Table 6 shows that our cross-cluster blocks based on self-attention between cluster tokens (S-A) outperforms the cross-attention based approach of Heidarianbaei et al. (2025) (C-A). The difference is relatively small on the SUM dataset (0.7% in $mF1$, 0.5% in OA); it is more pronounced on the CH dataset, with an increase of 3.4% in $mF1$ and of 2.8% in OA. We hypothesize that this difference between the datasets is related to the object size in relation to the resolution of the data. Some of the damage categories in the CH dataset occur in spatially small clusters that might vanish in the cross-attention scenario, where the face feature vectors are updated by weighted averages of all cluster features. In contrast, the SUM dataset consists of larger, urban-scale structures for which the impact of the global interactions might be limited.

Modality	OA [%]	$mF1$ [%]
Geometry	84.1	66.9
Texture	88.1	70.2
Both	94.3	**81.9**

Table 5. Comparison of quality metrics achieved on the SUM dataset when using different input modalities.

Method	SUM		CH	
	OA [%]	$mF1$ [%]	OA [%]	$mF1$ [%]
C-A	93.8	81.2	70.0	46.3
S-A (ours)	**94.3**	**81.9**	**72.8**	**49.7**

Table 6. Comparison of the results achieved on the two test datasets using two architectures of the TSTBs. C-A: global interactions are considered by cross-attention, following Heidarianbaei et al. (2025). S-A: our proposed method using self-attention in the cross-cluster blocks.

4.3.4 Comparison to other Methods: Table 7 presents the comparison of the results achieved by our method to the two baselines presented in Section 4.2.4. Figure 3 shows some qualitative results achieved by our method and the best-performing baseline. Our method outperforms these baselines by a large margin in all evaluation metrics. Compared to the best-performing baseline, NoMeFormer, the improvement in $mF1$ and OA is 15.3% and 11.7%, respectively, on the SUM dataset. On the CH dataset, the improvement in $mF1$ and OA achieved by our method over NoMeFormer is 15.6% and 4.9%, respectively. The results for DiffusionNet are still worse than those of NoMeFormer, with differences in $mF1$ larger than 20% on both datasets. We attribute the better performance of our method partly to its ability to leverage the rich textural information that cannot be fully captured through statistical texture descriptors alone. Figure 3 indicates that compared to NoMeFormer, our model produces notably sharper and more accurate predictions across all semantic classes. The results of NoMeFormer show a blurring effect, which we attribute to the properties of the global aggregation module used in (Heidarianbaei et al., 2025). The integration of cross-cluster blocks mitigates this problem, leading to a better balance between local and global features, although some fine-scale structures remain partially unresolved.

Method	SUM		CH	
	OA	$mF1$	OA	$mF1$
DiffusionNet	80.4	60.1	63.8	25.9
NoMeFormer	84.3	66.6	67.9	34.1
Ours	**94.3**	**81.9**	**72.8**	**49.7**

Table 7. Overall Accuracy (OA) and mean F1 score ($mF1$), both in [%], achieved by our method and two baselines on the SUM and CH datasets.

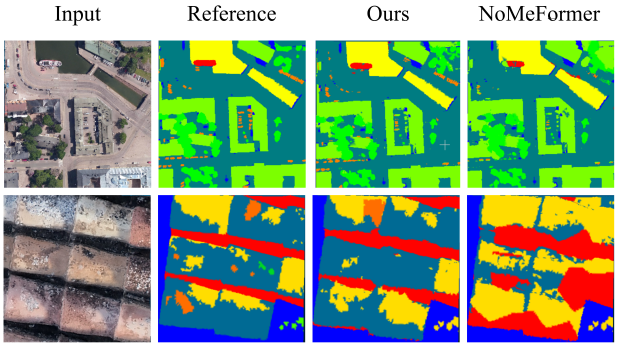

Figure 3. Qualitative results achieved by our method and by the best-performing baseline, NoMeFormer. From left to right: Input, reference, our method, and NoMeFormer. Top: an example from SUM. Bottom: an example from CH. Colour code — SUM: T (blue), V (dark green), Bld (green), W (yellow), C (orange), B (red). Colour code — CH: ND (greenish blue), J (red), St (green), D (chartreuse green), LI (gold), BC (yellow), SE (orange), unlabelled (blue).

Table 8 compares the results of different methods on the SUM dataset under the area-weighted evaluation protocol of (Gao et al., 2021), which prevents a disproportionate influence of densely sampled regions in datasets with large variations in triangle size. The results of DiffusionNet and NoMeFormer were achieved in our own experiments while those of the other methods are taken from (Gao et al., 2021).

Method	OA	$mF1$
RF-MRF	91.2	68.1
KPConv	93.3	76.7
DiffusionNet	81.0	61.3
NoMeFormer	84.9	68.6
Ours	**93.3**	**78.4**

Table 8. Area-weighted OA and $mF1$ scores [%] achieved by our method and four baselines on the SUM dataset.

As shown in the table, our method achieves the highest performance with a $mF1$ score of 78.4%, despite a marginal drop of 3.5% compared to the unweighted metric shown in Table 7. The fact that for NoMeFormer and DiffusionNet the area-weighted metrics are slightly better than the unweighted ones suggests that these methods are more effective on larger mesh faces, where local geometric context dominates. In any case, the performance gain of our method compared to these two baselines is still obvious. Compared to the best-performing methods according to (Gao et al., 2021), our method also performs better. It achieves the same OA as the point-based KPConv, but achieves a better area-weighted $mF1$ score (+1.7%), indicating the effectiveness of directly operating on mesh faces rather than relying on a point-cloud approximation, although the exact experimental protocol used in (Gao et al., 2021) is unclear. Compared to the mesh-processing framework RF-MRF, the area-weighted $mF1$ score achieved by our method is 10.3% higher; the improvement in OA is less pronounced, indicating even stronger problems for RF-MRF with underrepresented classes. The performance margin compared to RF-MRF indicates the advantage of deep learning for mesh-structured data.

5. Conclusion and Future Work

This work introduces a new transformer-based architecture for semantic segmentation of textured 3D meshes. In contrast to prior approaches that represent texture information by low-dimensional statistics, our model directly encodes texture through a transformer-based texture extraction branch, thereby preserving fine-grained appearance cues that are critical for distinguishing geometrically similar classes. Furthermore, a new approach to short- and long-range representation learning is proposed: the local mesh structure is captured through face-level processing, while global context is modelled via self-attention over cluster tokens, allowing to consider long-range interactions.

Extensive experiments performed on two datasets from different domains, including a new dataset for damage detection on a CH building, have demonstrated consistent performance gains of our method compared to existing ones. On the SUM benchmark, our model achieved a $mF1$ score of 81.9% and an OA of 94.3%, surpassing mesh- and point-based baselines alike. On the CH dataset, our method obtained a $mF1$ score of 49.7% and an OA of 72.8%, markedly improving over prior non-manifold mesh transformers. Ablations verify that replacing global cross-attention with cluster-token self-attention preserves fine-scale cues and improves performance ($+0.7\%$ on SUM and $+3.4\%$ on CH in $mF1$). These ablation studies confirm that incorporating raw texture information and enhancing global communication greatly improve mesh-based semantic segmentation.

Despite the promising results, several directions remain open. The most notable limitation of our method is the absence of explicit positional encoding in the texture transformer branch, which we leave for future exploration. In addition, memory and runtime constraints currently limit the number of faces and texture features that can be processed simultaneously, highlighting the need for more scalable attention mechanisms and model compression techniques. Additional performance gains may be achieved by integrating end-to-end learnable clustering, loss reweighting for class imbalance, and leveraging self-supervised pre-training on joint mesh geometry and texture. Extensions of additional modalities, including multispectral or thermal data, also represent compelling avenues for future development.

Acknowledgments

The research reported in this paper was performed in the context of the project ChemiNova (https://cheminova.eu/). ChemiNova has received funding from the European Union’s Horizon Europe Framework Programme under grant agreement 101132442.

References

Bishop, C. M., 2006. *Pattern Recognition and Machine Learning*. 1^{st} edn, Springer, New York (NY), USA.

Chu, X., Tian, Z., Wang, Y., Zhang, B., Ren, H., Wei, X., Xia, H., Shen, C., 2021. Twins: Revisiting the design of spatial attention in vision transformers. *Advances in Neural Information Processing Systems*, 34, 9355–9366.

Dong, Q., Wang, Z., Gao, J., Chen, S., Shu, Z., Xin, S., 2022. Laplacian2Mesh: Laplacian-based mesh understanding. *IEEE Transactions on Visualization and Computer Graphics*, 30, 4349-4361.

Feng, Y., Feng, Y., You, H., Zhao, X., Gao, Y., 2019. MeshNet: Mesh neural network for 3D shape representation. *AAAI Conference on Artificial Intelligence*, 33(1), 8279–8286.

Gao, W., Nan, L., Boom, B., Ledoux, H., 2021. SUM: A benchmark dataset of semantic urban meshes. *ISPRS Journal of Photogrammetry and Remote Sensing*, 179, 108-120.

Glorot, X., Bengio, Y., 2010. Understanding the difficulty of training deep feedforward neural networks. *13^{th} International Conference on Artificial Intelligence and Statistics*, 249–256.

Hanocka, R., Hertz, A., Fish, N., Giryes, R., Fleishman, S., Cohen-Or, D., 2019. MeshCNN: A network with an edge. *ACM Transactions on Graphics (TOG)*, 38(4), 1–12.

Hao, X., Wang, J., Leng, W., Zhang, R., Zhang, G., 2025. MFSM-Net: Multimodal feature fusion for the semantic segmentation of urban-scale textured 3D meshes. *Remote Sensing*, 17(9), 1573.

Heidarianbaei, M., Dorozynski, M., Mehltretter, M., Rottensteiner, F., 2025. NoMeFormer: Non-manifold mesh transformer. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, X-G-2025, 365–373.

Hu, Q., Yang, B., Xie, L., Rosa, S., Guo, Y., Wang, Z., Trigoni, N., Markham, A., 2020. RandLA-Net: Efficient semantic segmentation of large-scale point clouds. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11108–11117.

Hu, S.-M., Liu, Z.-N., Guo, M.-H., Cai, J.-X., Huang, J., Mu, T.-J., Martin, R. R., 2022. Subdivision-based mesh convolution networks. *ACM Transactions on Graphics (TOG)*, 41(3), 1–16.

Ioannidou, A., Chatzilari, E., Nikolopoulos, S., Kompatsiaris, I., 2017. Deep learning advances in Computer Vision with 3D data: A survey. *ACM Computing Surveys (CSUR)*, 50(2), 1–38.

Kölle, M., Laupheimer, D., Schmohl, S., Haala, N., Rottensteiner, F., Wegner, J. D., Ledoux, H., 2021. The Hessigheim 3D (H3D) benchmark on semantic segmentation of high-resolution 3D point clouds and textured meshes from UAV LiDAR and multi-view-stereo. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, 1, 11.

Lahav, A., Tal, A., 2020. Meshwalker: Deep mesh understanding by random walks. *ACM Transactions on Graphics (TOG)*, 39(6), 1–13.

Laupheimer, D., 2022. On the information transfer between imagery, point clouds, and meshes for multi-modal semantics utilizing geospatial data. PhD thesis, Institute of Photogrammetry and Remote Sensing, University of Stuttgart, Germany.

Loshchilov, I., Hutter, F., 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Masci, J., Boscaini, D., Bronstein, M. M., Vandergheynst, P., 2015. Geodesic convolutional neural networks on riemannian manifolds. *IEEE International Conference on Computer Vision Workshops (ICCVW)*, 832–840.

Maturana, D., Scherer, S., 2015. VoxNet: A 3D convolutional neural network for real-time object recognition. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 922–928.

Milano, F., Loquercio, A., Rosinol, A., Scaramuzza, D., Carlone, L., 2020. Primal-dual mesh convolutional neural networks. *Advances in Neural Information Processing Systems*, 33, 952–963.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017. PointNet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems*, 30.

Qian, G., Li, Y., Peng, H., Mai, J., Hammoud, H., Elhoseiny, M., Ghanem, B., 2022. PointNext: revisiting PointNet++ with improved training and scaling strategies. *Advances in Neural Information Processing Systems*, 35, 23192–23204.

Riegler, G., Osman Ulusoy, A., Geiger, A., 2017. OctNet: Learning deep 3D representations at high resolutions. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3577–3586.

Rouhani, M., Lafarge, F., Alliez, P., 2017. Semantic segmentation of 3D textured meshes for urban scene analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 123, 124–139.

Sharp, N., Attaiki, S., Crane, K., Ovsjanikov, M., 2022. Diffusionnet: discretization agnostic learning on surfaces. *ACM Transactions on Graphics (TOG)*, 41(3), 1–16.

Thomas, H., Qi, C. R., Deschaud, J.-E., Marcotegui, B., Goulette, F., Guibas, L. J., 2019. KPConv: Flexible and deformable convolution for point clouds. *IEEE/CVF International Conference on Computer Vision*, 6411–6420.

Tutzauer, P., Laupheimer, D., Haala, N., 2019. Semantic urban mesh enhancement utilizing a hybrid model. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W7, Copernicus GmbH, 175–182.

Vaswani, A., Shazeer, N. M., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. *Advances in Neural Information Processing Systems*, 30.

Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J., 2015. 3D ShapeNets: A deep representation for volumetric shapes. *IEEE Conference on Computer Vision and Pattern Recognition*, 1912–1920.

Yang, S., Hou, M., Li, S., 2023. Three-dimensional point cloud semantic segmentation for cultural heritage: a comprehensive review. *Remote Sensing*, 15(3), 548.

Zhang, X., Wang, H., Dong, H., 2025. A survey of deep learning-driven 3D object detection: Sensor modalities, technical architectures, and applications. *Sensors*, 25(12), 3668.

Zi, W., Li, J., Chen, H., Chen, L., Du, C., 2024. UrbanSegNet: An urban meshes semantic segmentation network using diffusion perceptron and vertex spatial attention. *International Journal of Applied Earth Observation and Geoinformation*, 129, 103841.