Exploring Point Transformers on 3D Semantic Segmentation of Javanese Architectures

Thodoris Betsas[1,2], Arnadi Murtiyoso[2], Pierre Grussenmeyer[2], Andreas Georgopoulos[1]

[1]National Technical University of Athens, School of Rural, Surveying and Geoinformatics Engineering, Lab of Photogrammetry - betsasth@mail.ntua.gr, drag@central.ntua.gr
[2]Université de Strasbourg, INSA Strasbourg, CNRS, ICube Laboratory UMR 7357, Photogrammetry and Geomatics Group, 67000, Strasbourg France - (theodoros.betsas, arnadi.murtiyoso, pierre.grussenmeyer)@insa-strasbourg.fr

Keywords: 3D Semantic Segmentation, Javanese Architecture, Point Clouds, Deep Learning, Cultural Heritage

Abstract

The complex geometry of Javanese architecture poses significant challenges for 3D semantic segmentation in cultural heritage documentation. This study evaluates state-of-the-art Point Transformers, i.e., PTv1, PTv2, PTv3, and LitePT, on the Sewu temple dataset, focusing on robustness and efficiency. While PTv1 and PTv2 achieve the highest Intersection-over-Union (mIoU 0.71), they incur high computational costs. Conversely, LitePT provides an optimal balance, delivering competitive results (0.69 mIoU) while being drastically faster. Furthermore, experiments with limited data reveal the significant benefits of transfer learning from European heritage datasets. We conclude that efficient Point Transformer architectures are promising for the automated understanding of complex non-European monuments.

1. Introduction

Semantic Segmentation is defined as the association of each element of the data under process, with a meaningful label. For instance, a meaningful label is associated with each (i) pixel, (ii) point, and (iii) triangle, when 2D images, 3D point clouds and 3D meshes are used, respectively. Nowadays, transformer based architectures dominate the 3D point cloud Semantic Segmentation (3DSS) domain in indoor, outdoor and hybrid environments, presenting strong generalization capabilities. Specifically, 3DSS transformer based architectures like PTv1 (Zhao et al., 2021), PTv2 (Wu et al., 2022), PTv3 (Wu et al., 2024), and LitePT (Yue et al., 2025) achieve high-end results in well-known 3DSS benchmarks such as Waymo (Sun et al., 2020), NuScenes (Caesar et al., 2020) and ScanNet++ (Yeshwanth et al., 2023).

In the cultural heritage (CH) domain 3DSS machine learning (ML) methods, like Random Forest (RF) and Support Vector Machines (SVMs), have been deeply investigated achieving high-end results. However, the ML methods, experience weak generalization capabilities, especially when applied to a diverse set of monuments. In fact, the CH monuments are characterized by complex architectural elements, which exhibit high heterogeneity and variability (Pierdicca et al., 2020; Betsas et al., 2025b), preventing traditional DL methods like PointNet (Qi et al., 2017a), PointNet++ (Qi et al., 2017b) and RandLa-Net (Hu et al., 2020), to learn meaningful representations (Tsarpalis, 2025) in demanding CH benchmarks like ArCH (Matrone et al., 2020).

In the context of CH monuments, 3DSS using DL architectures, typically relies on the DGCNN (Phan et al., 2018) model or its variants (Pierdicca et al., 2020; Cao and Scaioni, 2021). Moreover, there are recent applications investigating point transformers in the CH domain with promising results (Elalailyi et al., 2025; Zhang et al., 2025; Murtiyoso et al., 2025; Betsas et al., 2025b). However, point transformers have not been deeply investigated in the context of the 3DSS of CH monuments yet (Cao et al., 2022; Zhao et al., 2024; Betsas et al., 2025b) especially using Javanese Architectures. In general, the Javanese architectures constitute significant underrepresented heritage in 3DSS, compared to European datasets, posing an additional challenge regarding 3DSS.

Based on the above, the following research questions are posed:

- How robust is each point transformer version on 3DSS for Javanese architectures?
- Which are the factors that influence the performance of 3DSS using point transformers on Javanese architectures?
- Can pretrained models, on basically European datasets, improve 3DSS on Javanese architectures and under which conditions?

2. Preliminaries

In general, adapting 2D architectures to 3D space is not straightforward, due to the unique unstructured nature of point clouds (Betsas et al., 2025a). However, after the SotA results of transformers in natural-language-processing, and 2D scene-understanding tasks, Zhao et al. (2021) introduced one of the first transformer-based DL architectures for 3D scene understanding tasks, called Point Transformer (PTv1) and improved by a large margin the 3DSS performance in well-known benchmarks.

2.1 Point Transformer v1

Specifically, Zhao et al. (2021) proposed an encoder-decoder architecture, with skip-connections, using transition-down, transition-up and point-transformer blocks and a multi-layer-perceptron (MLP) head, for 3DSS. The encoder is constructed using multiple transition-down and point-transformer while the decoder using multiple transition-up and point-transformer blocks. Point-transformer layer is the core part of the point-transformer block, along with two linear layers and a skip-connection. Inside the point-transformer layer is the vector self attention operation applied on a local neighborhood of 3D points, defined using k-Nearest-Neighbors grouping (kNN).

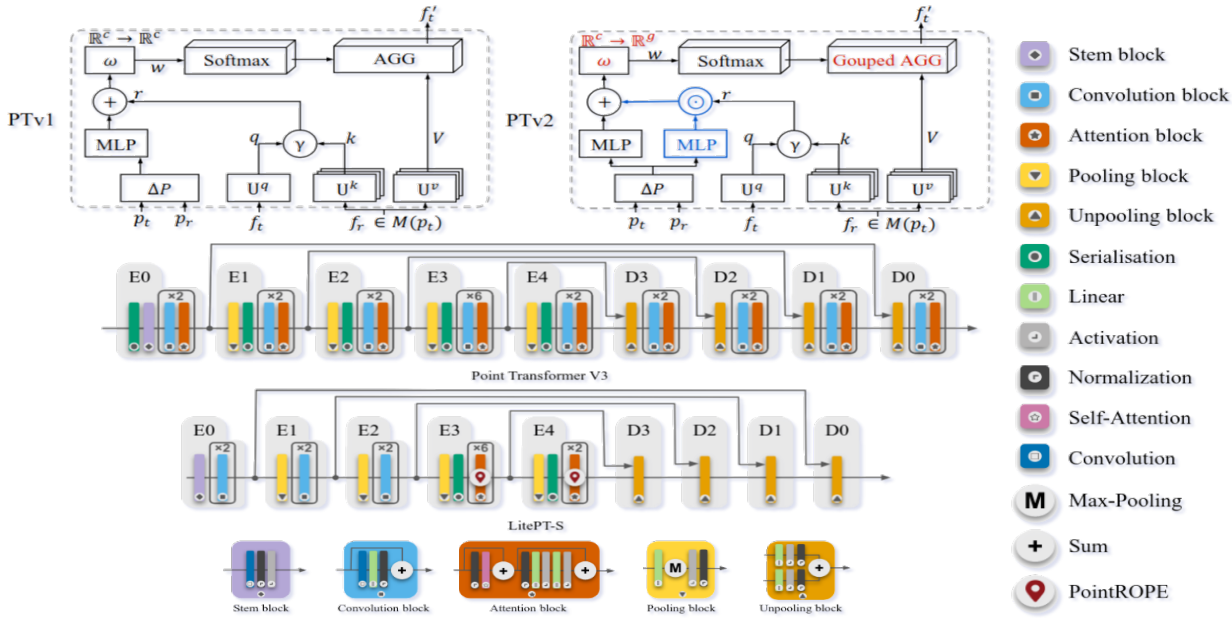

Figure 1. Attention mechanisms of PTv1, PTv2, PTv3 and LitePT, and PTv3, LitePT point transformer architectures. The attention mechanism and position encoding of PTv1 and PTv2, are adopted by Wu et al. (2022). The PTv3 and LitePT architectures and stem, convolution, attention, pooling and unpooling blocks are adopted by Yue et al. (2025)

The vector self-attention operation is defined by three linear projections, φ, ψ, α, a position encoding δ (MLP) and a mapping function γ (MLP). The linear projections, φ, ψ, α, constitute the Key, Query and Value matrices of the attention, respectively. The relation between Key and Query is the subtraction operation instead of dot product. The position encoding δ is defined as an MLP on the relative position of points. A detailed presentation of PTv1 attention mechanism and position encoding is presented in Figure 1. Furthermore, transition-down block is applied, on the output of the point-transformer block, using (i) farthest-point sampling (FPS), (ii) k-Nearest-Neighbors grouping (kNN), (iii) a trainable MLP and (iv) local-max-pooling operation. To be more specific, FPS is used to gather a uniform subset of the point cloud, kNN to form the set of k closest points to a given point using the dense representation, the MLP to increase the number of feature channels and local max-pooling to find the strongest response in each feature channel, for each point in the subset. Transition-down block expands the semantic depth of the point cloud while reduces its density.

Finally, transition-up block receives the semantically rich down-sampled point cloud of the encoder and the more detailed point cloud of the skip-connection for each stage. It then interpolates the rich features of the former with the geometric rich, and feature information of the later point cloud, using trilinear interpolation. Afterwards, it adds the upsampled with the skip-connection features and feeds the point transformer layer of the next stage. Finally, the MLP head is fed, to form the PTv1 output.

2.2 Point Transformer v2

Wu et al. (2022) argued that PTv1 is computationally and memory inefficient because it increases rapidly the number of channels and weights as it goes deeper, limiting it to shallow architectures and resulting in overfitting. PTv2 uses the same encoder-decoder architecture with skip-connections, as PTv1, but with three fundamental architectural shifts: (i) the grouped-vector-self-attention, instead of vector-self-attention, (ii) the partition-based pooling, instead of the transition-down block and (iii) a sophisticated position encoding multiplier mechanism. A detailed presentation of PTv2 attention mechanism and position encoding is presented in Figure 1.

To form the grouped-vector-self-attention, the channels of the Values matrix (features) are divided into groups, sharing the same weight, instead of the per-channel weights included in PTv1, reducing the number of trainable parameters. Additionally, in PTv1, the position encoding information is added to the Values matrix while in PTv2 this step is omitted. Moreover, the relative position encoding is processed by two parallel MLPs. The output of the relation function γ, is combined with the first MLP via the Hadamard product. Subsequently, the output of the second MLP is added to the combined features. The resulting tensor is then passed through the ω function, which consists of a Grouped Linear Layer followed by normalization and activation, before finally applying Softmax to produce the attention weights. These weights are then used in the Grouped Aggregation (Grouped AGG) layer to compute the weighted sum of the Value vectors. Finally, the pooling operation is applied using a pre-structured non-overlapping grid associated with a hash-table, indicating which 3D points belong to the same grid, reducing neighbor search time and replacing trilinear interpolation of PTv1, by a direct unpooling operation.

2.3 Point Transformer v3

Wu et al. (2024) observed that the breakthroughs in 2D deep learning methods are based on scaling principles like, the number of model parameters, the size of the receptive field and the dataset size, rather than of their intricate design. Furthermore, they stated that both PTv1 and PTv2 prioritize accuracy instead of efficiency. So they proposed PTv3, that consider the possibility of replacing complicated designs of certain mechanisms, enabling simplicity, efficiency and scaling in the context of point cloud processing. To achieve that, they replace the expensive neighbor-search (kNN) with point cloud serialization Figure 1),

using space-filling curves like Z-order and Hilbert. This mechanism serializes the unstructured 3D points into a contiguous 1D array based on the selected space-filling curve. In particular, every point in 3D space, is transformed to a 1D 64-bit integer based on their order on the space-filling curve. Every 64-bit integer begins with the batch index to which the point belongs, to enable batch processing. By sorting the points based on these integers, 1D neighbors become highly likely to be 3D spatial neighbors. Also the 1D representation of each batch is structured, meaning the neighbor points in 1D space are adjacent in memory. This property allows the authors to build on top of well-optimized and efficient methods, like window and dot-product attention, proposing patch attention. Each patch is defined using 1024 64-bit integers of the 1D representation following the order of different space-filling curves and thus increasing the receptive field diversely e.g., from 16 (kNN) neighbors to 1024.

Moreover, the patches interact with each other, using shuffle order, to form a global-sense of features. Apart, from kNN the authors state that the relative-positional-encoding (RPE) of PTv2 is inefficient, complex and time-consuming. In general, kNN neighbor search and RPE occupy 54% of the forward time of PTv2. Therefore, they introduced a sparse-convolution layer with skip-connection before attention, called xCPE to capture the local geometry. PTv3 follows an encoder-decoder structure (Figure 1) similar to PTv2, adopting grid-pooling. However, they introduced their serilized-attention mechanism, shuffle order and xCPE, and they replace batch-normalization with layer-normalization.

2.4 Lite Point Transformer

Yue et al. (2025) argued that convolution and attention operations can be used wisely, into the deep learning architecture, to improve efficiency. Notably, applying attention in early, high-resolution stages is computationally wasteful because local geometry can be captured more efficiently by convolutions. While applying attention in low-resolution deep layers can better capture the high-level semantics, context and global features. To delve more into this principle, the authors apply three variations of PTv3, in well-known benchmarks. The first variation is the original PTv3, the second and third ones are PTv3 without transformers and without sparse-convolution (SPConv), respectively. This experiment reveal that 67% of the learnable parameters in PTv3, are allocated to the sparse convolution layers of the positional encoding while the 30% of them account for the attention and MLP part. Also, the mIoU drop is larger when SPConv is omitted compared to the second variation. Furthermore, LitePT introduces Point-RoPE a training-free position-encoding method, based on rotations of the feature space, to replace the xCPE of PTv3. The authors extend the 1D RoPE implementation to 3D space, by grouping features according to the x, y and z axes and applying the standard 1D RoPE to each subspace. Point-RoPE is applied directly to the attention mechanism. Afterall, LitePT architecture use both SPConv and attention layers (Figure 1), resulting to a well-optimized architecture.

3. Methodology

3.1 Dataset

Following Murtiyoso et al. (2025) we re-use 3D surveying heritage data from the island of Java, Indonesia. Precisely, in 2019 the CIPA tropical school was organized in the Sewu temple complex in central Java (Murtiyoso et al., 2021), regarded as one of the biggest Buddhist religious complexes in Southeast Asia. The Sewu temple complex is constructed following a 100x100 m square mandala pattern. The main temple, was the House of Manjusri, who is the embodiment of Transcendent Wisdom (Prajna). Around the temple were a few hundred smaller shrines. In the exploited dataset there are 3 "Perwara" shrines and 2 medium-size guardians "Apit", temples, manually labeled using 11 labels (Figure 2, 3). The Sewu dataset is created using range data (TLS and LiDAR-UAV), and image data (aerial and oblique).

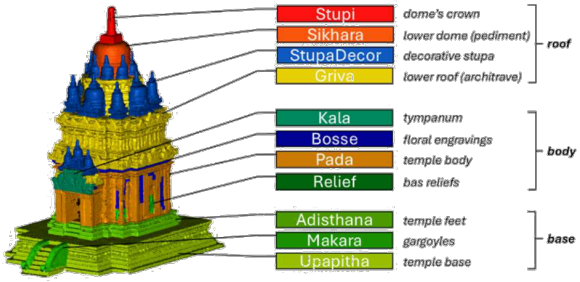

Figure 3. Classification of the temple architecture in the Sewu dataset (Murtiyoso et al., 2025)

3.2 Experiment Setup

In this paper, the 3DSS task is investigated regarding the Javanese architectures. In particular, the promising 3DSS results on the Javanese architectures (Murtiyoso et al., 2025) are enriched using the knowledge gained from the analysis of recent DL 3DSS methods on the ArCH dataset (Betsas et al., 2025b). Specifically, PTv1 excelled on the ArCH dataset, due to each detailed attention mechanism while PTv2 struggled to learn the detailed geometry of the ArCH monuments due to their high heterogeneity and variability. To further explore the ability of point transformers on CH data, we conducted a series of experiments investigating PTv1, PTv2, PTv3 and LitePT on Javanese Architectures using the Sewu dataset. The hardware specifications of the computing system used for training and testing of the point transformers algorithms, are presented in Table 1.

	CPU	GPU	RAM
Asus ROG Zephyrus G15	Ryzen 9 5900HS 3,3 GHz	NVDIA GeForce RTX 3070 (8GB)	40 GB DDR5

Table 1. Hardware specifications of the computing system used for training and evaluating the 3DSS models

Additionally, we adopted Pointcept (Pointcept-Codebase, 2025), an open-source code-base designed for point cloud perception tasks, that includes the original implementation of SotA 3DSS methods such as PTv1, PTv2 and PTv3. Furthermore, the recent LitePT algorithm has been developed based on the principles of Pointcept. Naturally, Pointcept does not include the necessary files to handle the Sewu dataset within its environment. Consequently, we generated the necessary configuration and preprocessing files required to apply point transformers on the Sewu dataset within Pointcept. In addition to configuration files, DL algorithms rely on numerous hyperparameters that must be carefully adjusted to achieve optimal results. Key parameters, such as learning rate, weight decay, and epoch count, are essential for training. Firstly, we train each point transformer, from scratch,

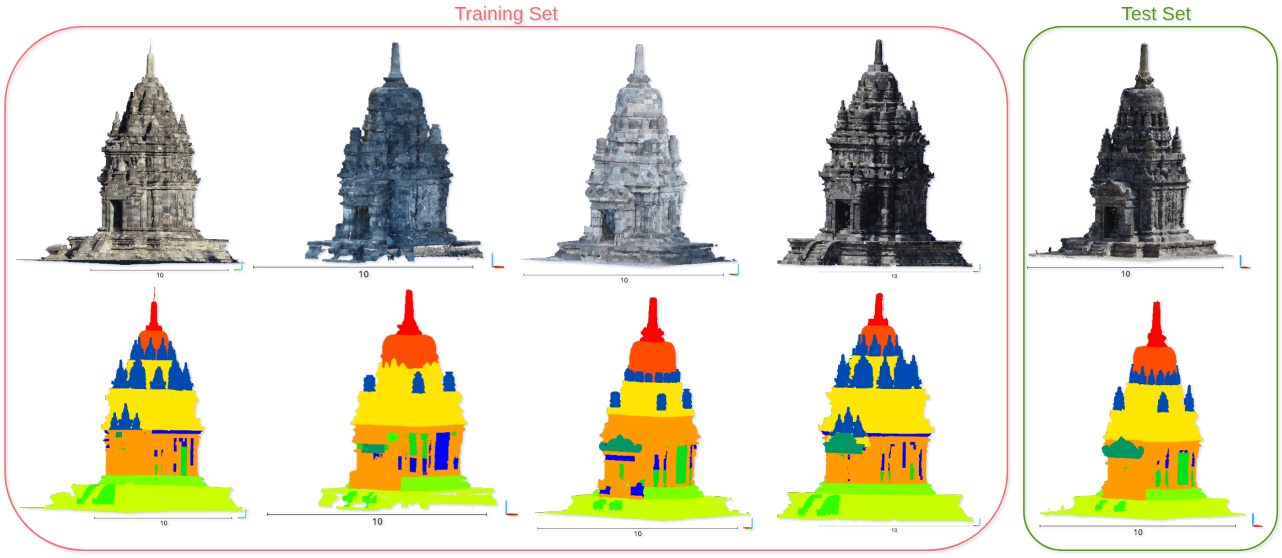

Figure 2. The Sewu Dataset. From left to right, Temples 1, 2, 3 and 5 (Training Set) and Temple 4 (Test Set)

using the training and test scheme presented in Figure 2. These experiments form the basis for the further investigation of each point transformer on the 3DSS of Javanese architectures. All the experiments were conducted using only the XYZ coordinates and RGB values i.e., excluding normals etc. A summary of the specific hyperparameters used for each point transformer model, in the base experiment, is provided in Table 2.

Algorithm	Epochs	Learning Rate	Weight Decay	Optimizer	Training Time(h)	Batch Size	Loss	Test mIoU
PTv1	100	0.0005	0.05	AdamW	21	1	Cross Entropy	0.7141
PTv2	100	0.0005	0,05	AdamW	16	1	Cross Entropy	0.7111
PTv3	100	0.0005	0.05	AdamW	23	1	Cross Entropy, Lovasz	0.6470
LitePT	100	0.0005	0,05	AdamW	19	12	Cross Entropy, Lovasz	0.6921

Table 2. Summary of the main hyperparameters and implementation details for each point transformer algorithm

The performance of the 3DSS algorithms is evaluated using qualitative and quantitative criteria. The former includes the subjective analysis of the result, using detailed visualizations of them, while the later includes the calculation of performance metrics. In 3DSS, the quantitative analysis is commonly include metrics like the Accuracy, Precision, Recall, F1-score, and Intersection over Union (IoU), among others. In this effort, the per-class Accuracy (Eq. 1), Precision (Eq. 2), Recall (Eq. 3), F1-score (Eq. 4) and IoU (Eq. 5) as well as their mean values are calculated. Of course, Pointcept includes the calculation of these metrics; however, we also evaluated the performance of each algorithm using SciKit Learn (SciKit-Learn, 2025) python module, for consistency among the compared methods.

$$OAcc = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 - score = 2\frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

$$IoU = \frac{TP}{TP + FP + FN} \tag{5}$$

where: TP = True Positives
TN = True Negatives
FP = False Positives
FN = False Negatives

Moreover, data augmentation was employed to enhance generalization, by introducing synthetic variability through transformations such as rotation, scaling, and jittering, an important step especially when the training data are limited. We utilized the standard augmentation pipelines provided within the Pointcept framework, adhering to the default configurations for each architecture without modifications. Table 3 details the specific augmentation techniques applied to each algorithm.

Data Augmentation	PTv1	PTv2	PTv3	LitePT
CenterShift	✓	✓	✓	✓
RandomScale	✓	✓	✓	✓
RandomFlip	✓	✓	✓	✓
RandomJitter	✓	✓	✓	✓
ChromaticAutoContrast	✓	✓	✓	✓
ChromaticTranslation	✓	✓	✓	✓
ChromaticJitter	✓	✓	✓	✓
GridSample	✓	✓	✓	✓
SphereCrop	✓	✓	✓	✓
CenterShift	✓	✓	✓	✓
NormalizeColor	✓	✓	✓	✓
RandomDropout			✓	✓
RandomRotate(x, y, z)			✓	✓
ElasticDistortion				✓

Table 3. Data augmentation techniques applied during training to PTv1, PTv2, PTv3 and LitePT models

4. Experiments

The performance of 3DSS DL methods is evaluated through both qualitative visualizations and quantitative metrics. The former constitute a subjective view of the results while the latter an objective view. This section presents the evaluation results for PTv1, PTv2, PTv3 and LitePT methods on the Sewu dataset.

4.1 Base Experiment

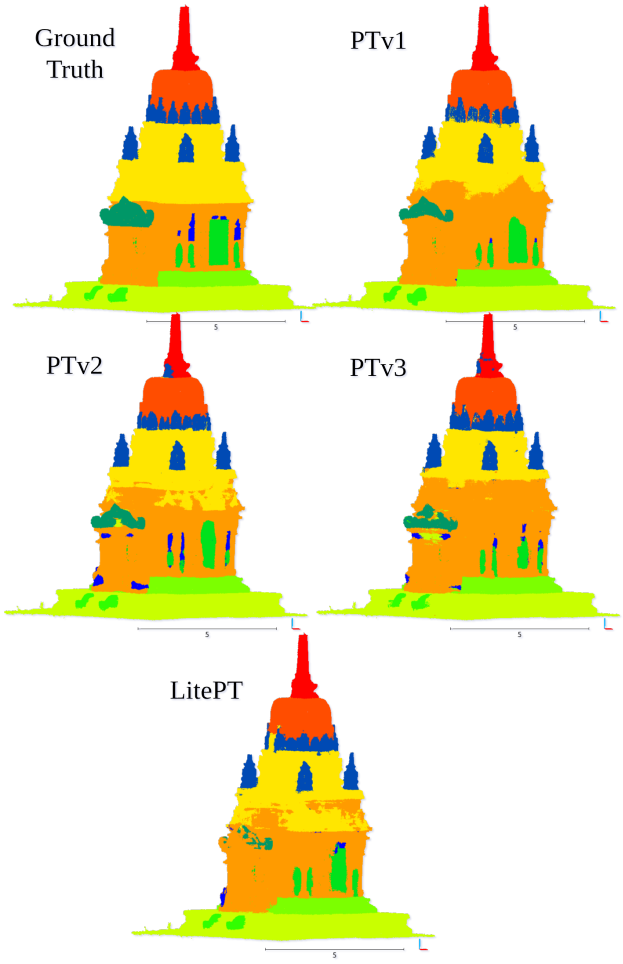

Figure 4. Qualitative Analysis on Temple_4 test scene for PTv1, PTv2, PTv3 and LitePT

In Figure 4 the predictions of PTv1, PTv2, PTv3 and LitePT on Temple_4 test set, are presented. Temples 1, 2, 3 and 5 are used for training using the hyperparameters presented in Table 2. Following the qualitative visualization, Tables 5 - 9 present the comparison among point transformer methods of accuracy, precision, recall, f1-score and IoU respectively, forming the quantitative analyses of the base experiment.

4.2 Limited data Experiment

Usually, in CH domain a large amount of training data are not available. In fact, DL methods are data hungry, especially when a supervised learning approach is used. In this context, ML methods are usually more convenient for 3DSS. To explore the performance of PTv1 under limited data we conducted different experiments using a part of the Sewu dataset. In Table 4 the limited data experiments are presented. Every training is for 100 epochs. In each experiment we change either the amount of data or the learning rate (LR).

Exp.	Training	Testing	LR	mIoU
	Trained from Scratch Limited Data			
1	Temples 2, 5	Temple 4	0.006	16.3%
	Pretrained Model ArCH Dataset Limited Data			
2	Temples 2, 5	Temple 4	0.006	24.2%
3	Temples 2, 5	Temple 4	0.0001	38.9%
4	Temples 2, 5	Temple 4	0.0005	**42.4%**
5	Temples 2, 5	Temple 4	0.00001	41.4%
	Trained from Scratch more Data			
6	Temples 2, 3, 5	Temple 4	0.0005	69.51%
7	Temples 1, 2, 3, 5	Temple 4	0.0005	**71.41%**

Table 4. Limited data experiments. LR: Learning Rate. mIoU on the Temple_4 test set.

5. Discussion

Cultural heritage monuments are characterized by complex architectural parts which exhibit high variability and heterogeneity among the different monuments. The experimental results in this study reveal a strong performance of PTv1, PTv2, PTv3 and LitePT using Javanese architectures. Notably, when sufficient number of data are used, point transformers achieve high-end results in both qualitative and quantitative perspective. During the experiments presented in Table 2, PTv1, PTv2, PTv3 and LitePT achieved 0.71, 0.71, 0.64, 0.69 mIoU, respectively. Additionally, the qualitative results depicted in Figure 4 reveal a strong performance of point transformers even in difficult classes like Kala and Stupa Decor. An example of their performance in demanding classes is presented in Figures 5 and 6. Apart from the case of Kala class in LitePT, the remaining examples indicate a similar performance among the point transformer algorithms. To this end the efficiency of the algorithm play a significant role.

As presented in Section 2 each point transformer algorithm is built on top of the previous one usually aiming to improve the efficiency while maintaining the high-performance, of its predecessor. Betsas et al. (2025b) stated that under high-variability monuments, e.g., ArCH benchmark, PTv1 achieved promising results while more recent algorithms like PTv2 didn't manage to learn strong features during training. This study shows that training on Javanese style temples, recent point transformers learned strong representations achieving similar performance to PTv1 in less training time (Table 2). Furthermore, the inference-time varies drastically, among the point transformer algorithms. In Table 10 the inference-time of each point transformer is presented. Specifically, the inference time reduced from 49 minutes, using PTv1 to 1.3 minutes using LitePT, while the performance drop is limited.

Algorithm	Inference (mins)	mIoU
PTv1	49	0.7141
PTv2	17	0.7111
PTv3	2	0.6470
LitePT	1.3	0.6921

Table 10. Inference-time for PTv1, PTv2, PTv3 and LitePT, on Temple 4 test scene

OAcc	Bose	Stupa Decor	Kala	Relief	Makara	Adisthana	Upapitha	Griva	Pada	Sikhara	Stupi	Mean
PTv1	**0.9961**	0.9840	0.9950	**0.9872**	0.9981	0.9845	0.9926	0.9310	0.9065	0.9916	**0.9998**	0.9788
PTv2	0.9906	0.9857	**0.9959**	0.9776	**0.9987**	0.9848	**0.9932**	0.9313	0.8938	0.9912	0.9981	0.9765
PTv3	0.9939	0.9862	0.9949	0.9694	0.9976	**0.9870**	0.9876	0.8938	0.8432	**0.9934**	0.9966	0.9676
LitePT	0.9900	**0.9867**	0.9900	0.9829	0.9986	0.9850	0.9931	**0.9546**	**0.9171**	0.9912	0.9996	**0.9808**

Table 5. Comparison of the Overall Accuracy of PTv1, PTv2, PTv3 and LitePT

Prec	Bose	Stupa Decor	Kala	Relief	Makara	Adisthana	Upapitha	Griva	Pada	Sikhara	Stupi	Mean
PTv1	**0.2880**	**0.9573**	0.9945	0.9497	0.8177	0.8415	**0.9859**	0.9519	0.6835	0.8773	0.9956	**0.8493**
PTv2	0.1921	0.9108	**0.9982**	0.9577	**0.9040**	0.8360	0.9835	0.9758	0.6578	0.8746	**0.9999**	0.8446
PTv3	0.1844	0.8579	0.9637	**0.9789**	0.8258	**0.8523**	0.9773	**0.9819**	0.5479	**0.9424**	0.9997	0.8284
LitePT	0.0293	0.9029	0.9853	0.9573	0.8869	0.8417	0.9830	0.9754	**0.7217**	0.9056	0.9998	0.8353

Table 6. Comparison of Precision of PTv1, PTv2, PTv3 and LitePT

Rec	Bose	Stupa Decor	Kala	Relief	Makara	Adisthana	Upapitha	Griva	Pada	Sikhara	Stupi	Mean
PTv1	0.0214	0.7606	0.6429	**0.7608**	**0.9221**	0.9379	0.9916	0.7292	**0.9303**	**0.9586**	**0.9936**	0.7863
PTv2	**0.4687**	0.8376	**0.7034**	0.5364	0.9066	0.9544	0.9960	0.7114	0.8992	0.9546	0.9020	**0.8064**
PTv3	0.1840	**0.9158**	0.6553	0.3414	0.8152	**0.9682**	0.9850	0.5364	0.9158	0.9189	0.8316	0.7334
LitePT	0.0518	0.8663	0.2874	0.6556	0.9034	0.9473	**0.9961**	**0.8180**	0.9047	0.9129	0.9795	0.7566

Table 7. Comparison of Recall of PTv1, PTv2, PTv3 and LitePT

F1	Bose	Stupa Decor	Kala	Relief	Makara	Adisthana	Upapitha	Griva	Pada	Sikhara	Stupi	Mean
PTv1	0.0398	0.8477	0.7809	**0.8448**	0.8668	0.8871	0.9888	0.8258	0.7880	0.9162	**0.9946**	0.7982
PTv2	**0.2725**	0.8727	**0.8253**	0.6877	**0.9053**	0.8913	**0.9897**	0.8229	0.7598	0.9129	0.9484	**0.8080**
PTv3	0.1842	**0.8859**	0.7801	0.5062	0.8204	**0.9066**	0.9811	0.6938	0.6857	**0.9305**	0.9079	0.7530
LitePT	0.0374	0.8842	0.4450	0.7782	0.8951	0.8914	0.9895	**0.8898**	**0.8029**	0.9092	0.9896	0.7738

Table 8. Comparison of F1-score of PTv1, PTv2, PTv3 and LitePT

	Bose	Stupa Decor	Kala	Relief	Makara	Adisthana	Upapitha	Griva	Pada	Sikhara	Stupi	Mean
PTv1	0.0203	0.7357	0.6406	**0.7313**	0.7649	0.7971	0.9778	0.7033	0.6502	0.8453	**0.9892**	**0.7141**
PTv2	**0.1578**	0.7741	**0.7025**	0.5240	**0.8270**	0.8039	**0.9796**	0.6990	0.6126	0.8397	0.9019	0.7111
PTv3	0.1015	**0.7952**	0.6395	0.3389	0.6955	**0.8291**	0.9630	0.5311	0.5217	**0.8700**	0.8314	0.6470
LitePT	0.0191	0.7925	0.2862	0.6370	0.8101	0.8041	0.9792	**0.8015**	**0.6707**	0.8335	0.9793	0.6921

Table 9. Comparison of IoU of PTv1, PTv2, PTv3 and LitePT

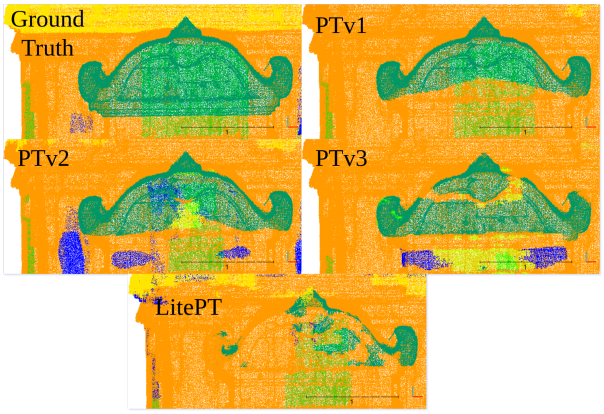

Figure 5. Close view for Kala class, for PTv1, PTv2, PTv3 and LitePT

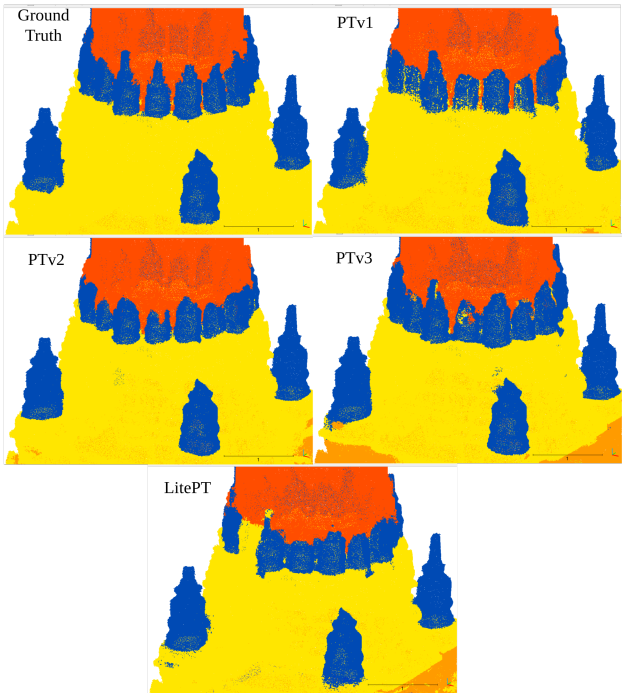

Figure 6. Close view for Stupa Decor class, for PTv1, PTv2, PTv3 and LitePT

However, PTv3 and LitePT are strongly connected on scale principle. Thus, we believe that if the training data increased the performance drop will be limited, revealing the strong performance of the recent point transformer algorithms. Additionally, based on Table 1, and the limited hardware (Table 2) used during training and inference, only LitePT is using batch of size 12 (bigger than 1), revealing its efficiency in respect to the other point-transformer algorithms.

Finally, in Table 4, a series of experiments using limited data is presented. Experiments 1 and 2, reveal that using the same hyperparameters and amount of data, the model that is trained on top of the best pretrained model on the ArCH dataset achieved better performance than the from-scratch trained model. Furthermore, by tuning the learning rate, the performance can be increased, from 24.2% to 42.4%. As it is expected, when more data are available the mIoU performance is increased drastically, from 42.4% to 71.41%.

6. Conclusion and Future Work

In this paper, the performance of recent point transformers is evaluated using Javanese architectures. In fact, point transformers dominate the 3DSS task in many well-known benchmarks. However, the evaluation of such algorithms on the CH domain is limited. To this end, the PTv1, PTv2, PTv3 and LitePT methods are evaluated on the Javanese architectures, aiming to asses their performance and find the factors that influence their generalization. Regarding, questions (i) and (ii) (Section 1), the amount of training data is crucial for training point transformers (Table 4). However, the performance of point transformers using Javanese architectures is promising (Figure 4 and Tables 5 - 9). Additionally, point transformers fine tuning regarding e.g., learning rate influences their performance drastically (Table 4). Regarding question (iii) (Section 1), point transformers performance is increased, especially using limited data, when a pretrained model on the ArCH dataset is used (Table 4). Moreover, the further analyses of PTv2, PTv3 and especially LitePT should be explored to deepen our understanding in the application of such methods in CH domain e.g., training the models on the ArCH dataset and fine-tuning them on the Javanese architectures. Additionally, a further investigation on the generalization of these models on other Asian temples, similar to the Javanese architectures, should be investigated. After all, the performance achieved by recent point transformers as long as their efficiency, in CH domain is promising.

References

Betsas, T., Georgopoulos, A., Doulamis, A., Grussenmeyer, P., 2025a. Deep Learning on 3D Semantic Segmentation: A Detailed Review. *Remote Sensing*, 17(2), 298.

Betsas, T., Tsarpalis, H., Georgopoulos, A., 2025b. Assessing Generalization Capability of 3D Semantic Segmentation Algorithms using 3D Point Clouds of Cultural Heritage. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 48, 111–118.

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O., 2020. nuscenes: A multimodal dataset for autonomous driving. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11621–11631.

Cao, Y., Scaioni, M., 2021. 3DLEB-Net: Label-Efficient Deep Learning-Based Semantic Segmentation of Building Point Clouds at LoD3 Level. *Applied Sciences*, 11(19). https://www.mdpi.com/2076-3417/11/19/8996.

Cao, Y., Teruggi, S., Fassi, F., Scaioni, M., 2022. A comprehensive understanding of machine learning and deep learning methods for 3d architectural cultural heritage point cloud semantic segmentation. *Italian Conference on Geomatics and Geospatial Technologies*, Springer, 329–341.

Elalailyi, A., Mazzacca, G., Alami, A., Padkan, N., Takhtkeshha, N., Fassi, F., Remondino, F. et al., 2025. 2d and 3d semantic segmentation for interpreting and understanding 3d heritage spaces. *Digital Heritage*, The Eurographics Association.

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. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 11108–11117.

Matrone, F., Lingua, A., Pierdicca, R., Malinverni, E. S., Paolanti, M., Grilli, E., Remondino, F., Murtiyoso, A., Landes, T., 2020. A benchmark for large-scale heritage point cloud semantic segmentation. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 43, 1419–1426.

Murtiyoso, A., Mazzacca, G., Remondino, F., Suwardhi, D. et al., 2025. Integrative AI for the Understanding of Ancient Javanese Architectures. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 48, 1065–1072.

Murtiyoso, A., Suwardhi, D., Grussenmeyer, P., Fadilah, W., Fauzan, K., Trisyanti, S., Macher, H. et al., 2021. Heritage documentation and knowledge transfer: A report on the CIPA Tropical School in Candi Sewu (Indonesia). *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 46, 493–497.

Phan, A. V., Le Nguyen, M., Nguyen, Y. L. H., Bui, L. T., 2018. Dgcnn: A convolutional neural network over large-scale labeled graphs. *Neural Networks*, 108, 533–543.

Pierdicca, R., Paolanti, M., Matrone, F., Martini, M., Morbidoni, C., Malinverni, E. S., Frontoni, E., Lingua, A. M., 2020. Point cloud semantic segmentation using a deep learning framework for cultural heritage. *Remote Sensing*, 12(6), 1005.

Pointcept-Codebase, 2025. Pointcept: A codebase for point cloud perception research. [Accessed 22-05-2025].

Qi, C. R., Su, H., Mo, K., Guibas, L. J., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 652–660.

Qi, C. R., Yi, L., Su, H., Guibas, L. J., 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30.

SciKit-Learn, 2025. scikit-learn: machine learning in Python &x2014; scikit-learn 1.6.1 documentation — scikit-learn.org. [Accessed 22-05-2025].

Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B. et al., 2020. Scalability in perception for autonomous driving: Waymo open dataset. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2446–2454.

Tsarpalis, H., 2025. 3d semantic segmentation using machine learning. Master's thesis, School of Rural, Surveying and Geoinformatics Engineering, National Technical University of Athens, Athens, Greece. Unpublished (In Greek).

Wu, X., Jiang, L., Wang, P.-S., Liu, Z., Liu, X., Qiao, Y., Ouyang, W., He, T., Zhao, H., 2024. Point transformer v3: Simpler faster stronger. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4840–4851.

Wu, X., Lao, Y., Jiang, L., Liu, X., Zhao, H., 2022. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems*, 35, 33330–33342.

Yeshwanth, C., Liu, Y.-C., Nießner, M., Dai, A., 2023. Scannet++: A high-fidelity dataset of 3d indoor scenes. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 12–22.

Yue, Y., Robert, D., Wang, J., Hong, S., Wegner, J. D., Rupprecht, C., Schindler, K., 2025. LitePT: Lighter Yet Stronger Point Transformer. *arXiv preprint arXiv:2512.13689*.

Zhang, M., Sun, S., Li, Z., 2025. An Automatic Measurement Method for Architectural Heritage Based on Point Cloud Semantic Segmentation Algorithm: A Case Study of the Hollow Watchtowers of the Ming Great Wall. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 48, 1691–1698.

Zhao, H., Jiang, L., Jia, J., Torr, P. H., Koltun, V., 2021. Point transformer. *Proceedings of the IEEE/CVF international conference on computer vision*, 16259–16268.

Zhao, J., Liu, R., Hua, X., Yu, H., Zhao, J., Wang, X., Yang, J., 2024. DSC-Net: learning discriminative spatial contextual features for semantic segmentation of large-scale ancient architecture point clouds. *Heritage Science*, 12(1), 274.