Contents lists available at ScienceDirect

Computers & Graphics

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

OrthoCAD-322K: A cross-modal approach for retrieving 3D CAD models from orthographic views using a graph-based framework on a developed large-scale dataset

ARTICLE INFO

Article history:
Received January 25, 2026

2D to 3D CAD retrieval, Orthographic views, Cross-modal learning, Graphormer, Engineering drawing dataset, Industrial CAD retrieval

ABSTRACT

Despite the widespread adoption of 3D CAD systems, 2D orthographic drawings remain integral to engineering workflows. However, millions of legacy drawings lack corresponding 3D models, hindering their integration into modern simulation, manufacturing, and digital twin systems. Existing methods for 2D to 3D CAD retrieval often fall short of meeting the structural precision required for engineering-grade drawings.

We propose a cross-modal retrieval framework that aligns vector-based 2D DXF (Drawing Exchange Format) views with 3D CAD models using contrastive learning. Our architecture integrates a Graphormer-based encoder for 2D input and a PointNet-based encoder for 3D CAD models. We introduce a novel proximity-based spatial encoding to enhance structural precision and robustness across varying view configurations. Using the filtered subset (~283K) of the newly developed large-scale dataset OrthoCAD-322K, extensive ablation and comparison studies demonstrate the robustness and generalization of the model in different input conditions and architectures.

1. Introduction

Computer-Aided Design (CAD) systems play a fundamental role in engineering, with applications that span mechanical, aerospace, automotive, and architectural domains. Historically, engineering workflows have relied on 2D orthographic projections, such as front, top, and side views, for tasks that include manufacturing, technical documentation, and communication. These projections are commonly stored as DXF files (Drawing Exchange Format), a vector-based CAD data format that facilitates interoperability across design platforms. Despite earlier expectations that solid modeling systems would replace traditional drafting systems, 2D CAD applications remained dominant even into the mid-2000s [1]. Before the widespread adoption of 3D modeling tools, early CAD systems primarily produced 2D drawing DXF files, resulting in extensive repositories of legacy engineering data.

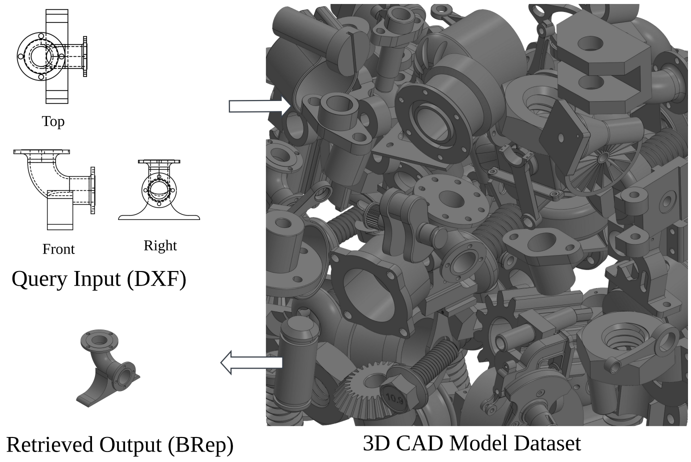


Fig. 1. Retrieval of 3D CAD model (BRep) using orthographic (DXF) query.

A significant proportion of these drawing files lack corre-

[*]Corresponding author: Tel.: +0-000-000-0000; fax: +0-000-000-0000;
e-mail: example@email.com (), example@email.com (Corresponding Author Name)

Table 1. Comparative analysis of existing 2D–3D CAD datasets in terms of sample size, CAD model types, 2D view formats, view configurations, and supported geometric primitives. Our proposed OrthoCAD-322K dataset offers significantly greater scale, standardization, and support for full orthographic view projection compared to prior datasets, including Automatic 3D CAD Reconstruction [2], CAD2Program [3], PlankAssembly [4], RL-Based CAD Reconstruction [5], Photo2CAD [6], SPARE3D [7], and Text2CAD [8].

Aspect	Automatic 3D CAD Reconst.	CAD2PROGRAM	PlankAssembly	RL-Based CAD Reconstruction	Photo2CAD	SPARE3D	Text2CAD	Ours
Total Samples	2.9K	368K	26.7K	68.9K	Not specified	21.5K	100K	322K
CAD Model Type	B-Rep	3D customized cabinet models	Cabinet furniture	B-Rep	CSG-based	B-Rep and CSG-based	B-Rep	B-Rep
2D View File Format	Vectorized (SVG)	Rasterized	Vectorized (SVG)	Vectorized (SVG)	Photos (PDF, DXF, and scanned images)	Vectorized (SVG) and Rasterized	Vectorized (SVG) and Rasterized	Vectorized (DXF), Rasterized and SolidWorks Native (SLDDRW)
Views Provided	Front, Bottom, Left	Front, Side, Top, Section views	Front, Top, Side	Front, Bottom, Left	Front, Top, Side	Isometric + Front, Top, Right	Isometric + Front, Top, Side	Isometric + Front, Back Top, Bottom, Left, Right + Projected views for each Orthographic View
Primitive Types	Solid Mechanical parts	373 cabinet primitives	Axis-aligned cuboids (planks)	General mechanical primitives	Simple Geometric Primitives	Solid Mechanical parts	General mechanical components	General mechanical components

sponding 3D models [6] due to the limitations of contemporary CAD tools and the absence of formalized 3D modeling practices during earlier design stages. This lack of 3D geometry presents several challenges: (1) manual reconstruction from drawing files is time-consuming and error-prone [9], often requiring expert interpretation; (2) the absence of 3D models limits the use of modern engineering tools such as finite element analysis (FEA) [10], digital twin simulation, and computer-aided manufacturing (CAM); and (3) design reuse is hindered, complicating integration into contemporary Product Lifecycle Management (PLM) systems.

Although 3D reconstruction from 2D orthographic views has been extensively explored, the majority of existing methods remain rule-based or heuristic. Several approaches [9] typically assume complete and clean input, and they often struggle with complex geometric features such as splines [2], nested loops, and topological ambiguities. Moreover, they are highly sensitive to real-world noise. In practical scenarios, engineering drawings frequently exhibit inconsistent view configurations and missing geometric entities (e.g., lines, arcs, splines).

To address these challenges, recent research has introduced deep learning-based methods for reconstructing 3D models from 2D orthographic views. However, existing deep learning approaches often target narrow applications such as cabinet or furniture design [3, 4]. These methods often fail to handle essential drawing elements such as spline curves [11] and typically output unstructured point clouds rather than editable CAD models. Moreover, some are limited to primitive shapes [6] or support only basic modeling operations such as extrusion [5].

Furthermore, many of these methods are trained on proprietary [3] or narrowly spanning datasets that are often limited in geometric diversity, complexity, or scale. As summarized in Table 1, the existing datasets vary significantly in fidelity and structural completeness, restricting the development of scalable and generalizable 2D to 3D learning systems. Additionally, while most prior work focuses on direct 3D reconstruction, the alternative task of cross-modal retrieval, retrieving a structurally relevant 3D CAD model from one or more orthographic views, remains comparatively underexplored. To address this gap, we introduce OrthoCAD-322K the first large-scale dataset with standardized orthographic and isometric views for 322,000 publicly available CAD models in both DXF and raster formats.

Retrieving 3D models from 2D orthographic views presents its unique set of challenges. These 2D views, while geometrically precise, lack depth cues, are susceptible to viewpoint dependency, and often exhibit hidden line ambiguities [12, 13]. Critical engineering features such as holes, ribs, fillets, and chamfers may be partially or fully hidden in individual projections, requiring multiview synthesis for accurate interpretation.

In this work, we address the problem of retrieving 3D CAD models from a variable number of orthographic views stored in DXF files, which may be incomplete or contain partially missing geometric information (Fig. 1). We propose a deep learning framework that bridges the modality gap between 2D DXF drawings and 3D geometry by aligning them in a shared embedding space. Our method employs cross-modal contrastive learning, enabling robust retrieval. A key enabler of our approach is the use of vector-based DXF files, which represent lines, arcs, and splines with high fidelity and support the extraction of geometric and topological features [14]. Recent work shows that vector-based graph attention networks outperform raster-based methods in segmenting complex technical drawings [15].

To encode 2D geometry, we adapt the Graphormer [16] architecture by introducing a novel proximity-based spatial encoding specifically designed for CAD DXF views. This strategy connects nodes based on their spatial closeness, enabling efficient capture of local geometric relationships while avoiding the high computational cost of shortest-path calculations in dense graphs. For the 3D modality, we sample CAD surfaces into point clouds and project them onto the faces of an Oriented Bounding Box (OBB), which are then processed by a PointNet [17] encoder. The resulting 3D features are jointly aligned with 2D view features in a shared embedding space. Our approach achieves strong retrieval performance, reaching 94.51% top-1 accuracy with three orthographic views (front, top and right), and remains robust even under incomplete input conditions.

This paper makes the following contributions:

- Developed the OrthoCAD-322K dataset comprising standardized orthographic and isometric views for 322,000 publicly available BRep CAD models.
- Proposed a novel cross-modal retrieval framework that aligns 2D DXF orthographic views with 3D CAD models via contrastive learning.
- Designed a novel proximity-based spatial encoding for 2D views within the Graphormer encoder, offering a more efficient alternative to shortest-path encodings.

2. Related Work

The retrieval of 3D CAD models from 2D representations has evolved significantly over several decades, with various approaches that address the inherent challenges of cross-modal matching and geometric understanding. Early CAD retrieval used global descriptors, structural encodings, and topology graphs, but lacked fine-grained discrimination, leading to learning-based methods [18]. Recent work has also explored structured formats like STEP for both classification and retrieval tasks, utilizing Graph Neural Networks to learn from the format's rich topological and semantic data [19].

2.1. View Mapping and Multiview Fusion Techniques

Various techniques have addressed the 2D–3D modality gap through view mapping and multiview reasoning, including methods for cross-domain alignment, multiview projection with canonical viewpoints, and feature aggregation strategies using CNNs, graph networks and attention mechanisms [18]. Other approaches utilize graph neural networks [20] and multilayer perceptrons [21] to jointly encode multiview structure. Although effective, most methods target rasterized renderings or sketch inputs and often lack support for the geometric fidelity and symbolic structure required in engineering grade CAD applications [18].

2.2. Sketch-Based 3D CAD Retrieval

Retrieval methods based on sketches of orthographic views has been a prominent approach to bridge 2D and 3D representations. Liu et al. [22] introduced a user-adaptive sketch-based CAD retrieval system using statistical modeling of individual drawing habits. Pu and Ramani [23] developed a 2D sketch-based interface that requires users to draw three orthographic views for reliable retrieval. Wang et al. [24] used a hybrid approach combining geometric outlines and skeletal topologies.

To address inaccuracies in query sketches, SketchCleanNet [25] was introduced as a learning-based approach to improve retrieval performance by denoising and correcting overdrawn or incomplete sketches. Building on this, SketchCleanGAN [26] used adversarial learning to further refine or complete defective sketches, showing improved robustness in large-scale retrieval scenarios. These methods leverage the CADSketchNet dataset [27], which includes annotated and synthetic hand-drawn sketches for engineering components.

Other approaches such as Wang and Zhou [28] have employed convolutional neural networks and contrastive learning to create joint embeddings between sketches and 2D renderings of 3D shapes for zero-shot retrieval. Su et al. [29] and Qi et al. [30] extended this paradigm with multiview CNNs for 3D shape recognition.

2.3. Datasets for 2D to 3D CAD Mapping

Existing datasets for 2D to 3D CAD reconstruction vary in scale, geometric diversity, and modeling fidelity. As shown in Table 1 datasets such as SPARE3D [7] contain limited collections of synthetic CAD and CSG models that lack real-world complexity. The Automatic 3D CAD Reconstruction dataset [2], derived from Fusion360, excludes B-spline surfaces and therefore lacks complex freeform geometries.

Domain-specific datasets like PlankAssembly [4] and CAD2Program [3] focus solely on cabinet furniture, which restricts their applicability to broader CAD tasks. PlankAssembly [4] is further constrained to axis-aligned cuboids as its primary primitive type, excluding curved geometries.

The reinforcement learning framework in [5] is trained on a dataset of loop-path pairs extracted from 2D orthographic drawings, enabling parametric CAD reconstruction primarily through extrusion operations. GaussianCAD [11] provides a dataset focused on 3D Gaussian Splatting for CAD reconstruction from three orthographic views. A notable recent contribution, Text2CAD [8] introduces isometric and orthographic (front, top, side) views as intermediates for text-to-CAD generation across a dataset of 100,000 models. Benchmark BRep datasets such as ABC [31], DeepCAD [32], and Fusion360 [33] provide rich parametric CAD models with high geometric fidelity. However, they lack associated 2D orthographic views.

3. Overview of the Proposed Approach

As illustrated in Fig. 2, this work presents a cross-modal retrieval framework that matches 2D orthographic DXF queries to 3D CAD models. The system operates directly on structured vector-based views (DXF) and employs a dual encoder architecture trained with contrastive learning.

To capture structural relationships, we introduce a novel proximity-based spatial encoding. This connects entities based on geometric closeness, allowing the model to learn spatial context and view alignment without relying on computationally expensive shortest-path calculations. This encoding is embedded within a Graphormer based encoder that models global interactions across views.

In parallel, 3D CAD models are converted from STEP files to surface-sampled point clouds. These are projected onto six canonical planes and encoded using a PointNet [17]-based network. The 2D and 3D encoders are trained jointly with contrastive loss, bringing matched 2D and 3D representations closer in a shared embedding space.

The key novelty of our approach lies in aligning these heterogeneous modalities, vectorized 2D views, and 3D point clouds, within a shared embedding space. By leveraging vector geometry and spatially-aware encoding, the framework achieves high retrieval accuracy (94.51% top-1 with three views: front, top and right) and maintains robustness under partial or degraded input, making it well suited for real-world industrial drawings.

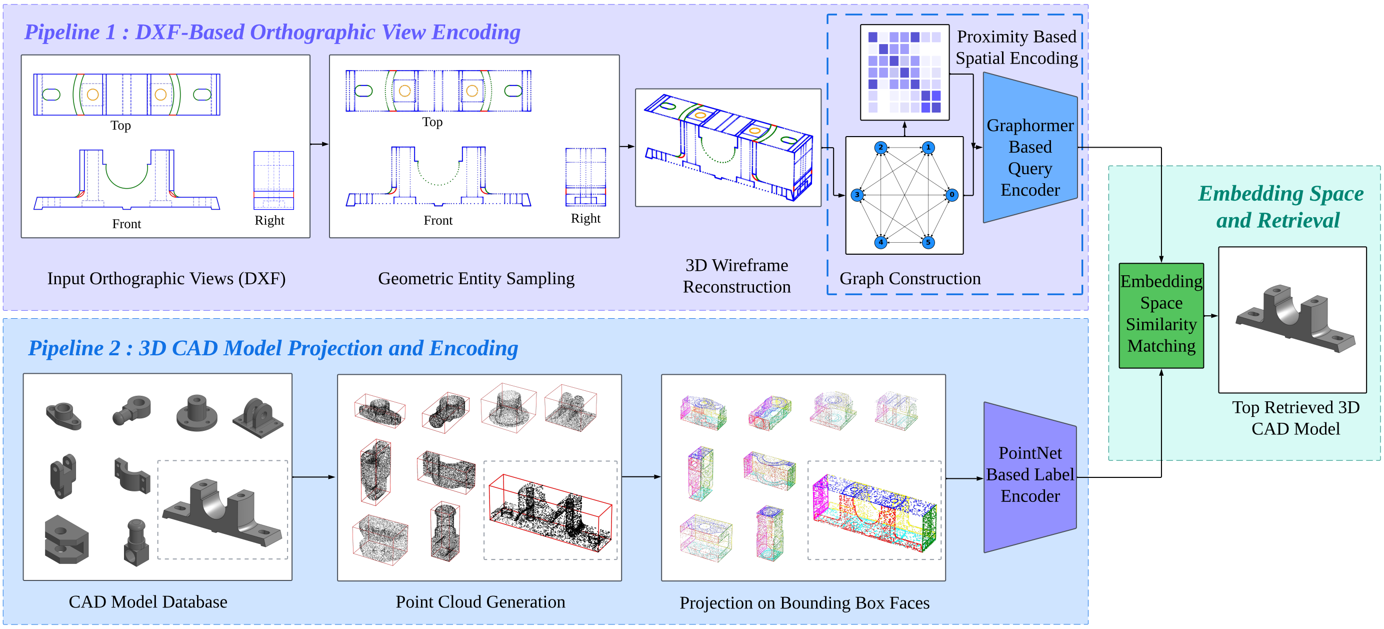

Pipeline 1 : DXF-Based Orthographic View Encoding
Top
Front
Right
Input Orthographic Views (DXF)
Top
Front
Right
Geometric Entity Sampling
3D Wireframe Reconstruction
Proximity Based Spatial Encoding
Graph Construction
Graphormer Based Query Encoder
Pipeline 2 : 3D CAD Model Projection and Encoding
CAD Model Database
Point Cloud Generation
Projection on Bounding Box Faces
PointNet Based Label Encoder
Embedding Space and Retrieval
Embedding Space Similarity Matching
Top Retrieved 3D CAD Model

Fig. 2. Overview of the proposed Cross-Modal Retrieval Framework from 2D Orthographic views to 3D CAD Models.

4. Data Generation and preprocessing

Our data generation process implements a pipeline for transforming CAD models (STEP) into 2D standardized representation format. We integrate three CAD model repositories as source data. Fusion360 [34, 33], DeepCAD [32], and CAD-Parser [35]. Image-based hashing techniques are used for duplicate elimination by capturing screenshots of 3D models from multiple viewpoints.

4.1. Orthographic View Generation

We implemented an automated pipeline to generate standardized 2D views from 3D STEP models using SolidWorks Task Scheduler. Each CAD model was converted to the SolidWorks native format (`.sldprt`) and rendered into one isometric and six orthographic views (front, back, top, bottom, left, right) based on a third-angle projection standard. A predefined drawing template ensured consistent layout and projection parameters across all samples.

The resulting drawings (`.slddrw`) were exported in both DXF (vector) and raster formats for downstream processing. Fig. 3 summarizes the overall view generation and file conversion workflow.

4.1.1. Data Cleaning and Generation Infrastructure

To further improve the quality of the dataset, a lightweight graphical user interface (GUI) was developed to support manual inspection. A group of senior mechanical engineering undergraduates reviewed the dataset and identified two recurring issues: (i) Views collapsing into lines or points due to non-volumetric models and (ii) scaling errors where projections extended beyond drawing sheet boundaries. A total of 7.4K views affected by issue (i) were excluded from the dataset. For issue (ii), instead of discarding the affected BRep models (1.3K in total), we refined the generation pipeline to apply uniform scaling, normalizing BRep into a consistent spatial extent prior to view generation.

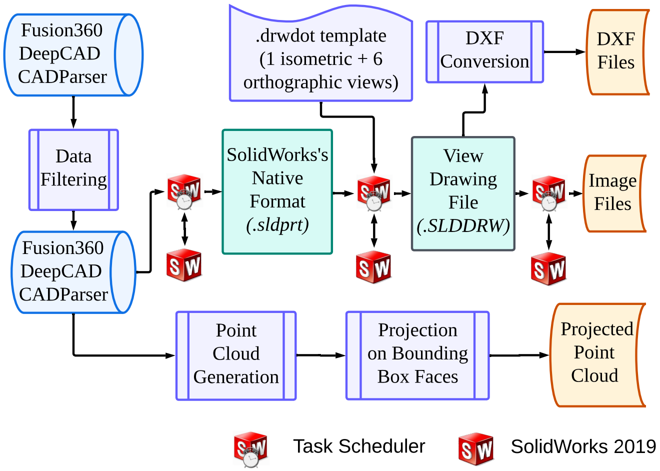


Fig. 3. Framework for Multimodal CAD data generation.

The complete dataset was generated over a span of three months for the initial full run using 12 parallel instances of SolidWorks with shared storage. This was followed by an additional two-week post-processing phase to handle failed or restarted cases. In total, DXF and rasterized views were created for 322,000 BRep CAD models. As illustrated in Fig. 4, the resulting OrthoCAD-322K dataset offers rich geometric diversity, with each sample comprising paired 2D views and corresponding 3D models.

4.2. DXF Preprocessing

Preprocessing begins by parsing DXF files to extract geometric entities (lines, arcs, circles, splines, polylines, lwpolylines (lightweight polyline)) across six canonical orthographic views, formally denoted as $v_i \in V = \{v_0, \ldots, v_5\}$, where v_0 = front, v_1 = back, v_2 = left, v_3 = right, v_4 = top, and v_5 = bottom. Our framework currently supports only geometric entities in DXF files; annotations such as dimensions, tolerances, or symbols must be removed through preprocessing.

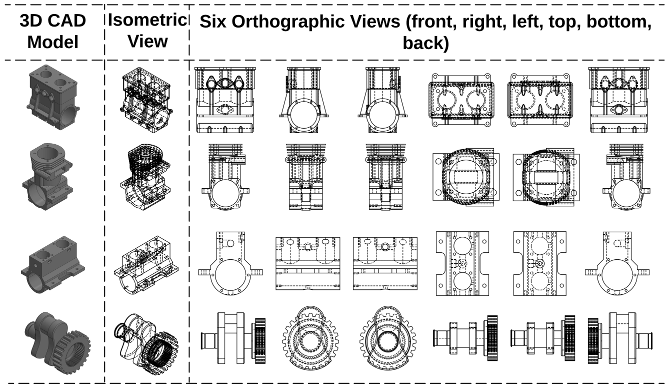

3D CAD Model
Isometric View
Six Orthographic Views (front, right, left, top, bottom, back)

Fig. 4. Sample from the Multimodal CAD Dataset comprising CAD models (BRep), isometric view, and six standard orthographic views (front, back, left, right, top, bottom), provided in both DXF (vector) and rasterized image formats for each view.

To ensure computational feasibility and comply with the constraints of the Graphormer encoder, we filtered out DXF samples in which any single view among the six orthographic projections contained more than 100 geometric entities. This reduced the dataset from 322,000 to ~283,000 CAD models (≈ 12.11% excluded). This filtered subset of ~283,000 models was used for all retrieval training and evaluation experiments reported in this work.

4.2.1. *Geometric Entity Sampling Methodology*

Adaptive sampling is an adaptive point-allocation strategy that converts each geometric entity into a fixed-size point set by allocating sample points according to local geometric importance, concentrating them near endpoints and high-curvature regions to improve shape-representation accuracy (see Fig. 5 A). This method enhances shape representation compared to uniform sampling, while keeping the number of sample points per entity fixed.

Each geometric entity is sampled into 100 points adaptively. Type-specific strategies are used to preserve the overall shape. For a given entity e, the resulting set of sampled points is defined as $P_e = \{p_1, p_2, \ldots, p_{100}\} \subset \mathbb{R}^2$.

The Sampling methodology varies with geometric entity type:

- *Lines* are discretized using adaptive cosine-based interpolation that strategically biases point distribution toward endpoints:

$$t_i = 0.5(1 - \cos(\pi i/99)), \quad p_i = (1 - t_i)p_{\text{start}} + t_i p_{\text{end}} \tag{1}$$

- *Arcs* undergo adaptive angular interpolation with similar endpoint emphasis:

$$\theta_i = \theta_s + (\theta_e - \theta_s) \cdot \frac{1 - \cos(\pi i/99)}{2}, \quad p_i = c + r(\cos\theta_i, \sin\theta_i) \tag{2}$$

- *Circles* utilize uniform angular sampling. $\theta_i = 2\pi i/100$
- *Polylines* and *lwpolylines* are sampled using a combination of straight and curved segments to capture their overall shape.
- *Splines* are sampled adaptively to capture geometric detail, concentrating points in high-curvature regions[1] [36].

4.2.2. *Wireframe Reconstruction Process*

The 2D points sampled from each view are lifted into 3D using view-dependent transformation functions. If view labels are missing, they can be inferred using geometric heuristics. The view with the largest projection area and most visible features is typically identified as the front view. Other views are matched by comparing shared dimensions, such as height or width, with the identified front. For each view $v_i \in V$, a transformation $T_i : \mathbb{R}^2 \to \mathbb{R}^3$ maps the 2D coordinates to their 3D positions, as shown in Fig. 5 B:

$$T_i(p) = \begin{cases} (x, 0, y) & \text{if } v_i \in \{\text{front}, \text{back}\}, \\ (0, x, y) & \text{if } v_i \in \{\text{left}, \text{right}\}, \\ (x, y, 0) & \text{if } v_i \in \{\text{top}, \text{bottom}\}, \end{cases} \tag{3}$$

Here, T_i map the 2D points of view $v_i \in V = \{v_0, \ldots, v_5\}$ to their corresponding 3D coordinates.

Following the transformation, the 3D entities of all views are merged to construct a unified 3D wireframe graph $G = (N, E)$, where each node $n \in N$ corresponds to a sampled entity and encodes a feature vector:

$$f_n = [o_v, o_e, v_b, \text{vec}(P_e)] \tag{4}$$

where:

- $o_v \in \{0, 1\}^6$ is a one-hot encoding of the view type,
- $o_e \in \{0, 1\}^6$ is a one-hot encoding of the entity type,
- $v_b \in \{0, 1\}^4$ is the visibility flag vector representing occlusion status,
- $\text{vec}(P_e) \in \mathbb{R}^{300}$ is the flattened sampled point cloud.

The visibility vector v_b is represented using a 4D one-hot encoding, with $[0.0, 1.0, 0.0, 0.0]$ indicating visible entities and $[0.0, 0.0, 0.0, 1.0]$ indicating hidden entities. The 4D format is designed to ensure that visibility features contribute meaningfully during training, despite the presence of 300 geometric features. This helps the model attend to visibility cues effectively.

Edges are formed between pairs of nodes based on spatial proximity, and each edge $e_{ij} \in E$ is assigned attributes a_{ij} (edge attributes) as:

$$a_{ij} = \left[1 - \frac{d_{ij}}{d_{\max}}, d_{ij}, c_i, c_j\right] \tag{5}$$

where d_{ij} is the Euclidean distance between the centers c_i and c_j of entities i and j, and $d_{\max}$ is the maximum pairwise distance observed used for normalization.

To support stable learning, the graph is normalized. All 3D points are shifted so the graph is centered at the origin. It is then scaled so its largest dimension is one unit. This makes the graph scale-invariant and consistent across samples. The final graph is saved in PyTorch Geometric format (`.pt`).

4.3. *Data preprocessing for STEP file labels*

Each 3D CAD model is first converted from STEP format to a triangulated mesh (OBJ) representation. To obtain the surface point cloud from each mesh, we use a deterministic, area-weighted sampling strategy. Specifically, points are sampled over triangle surfaces in proportion to their area using barycentric interpolation with a fixed seed. This prioritizes sampling in curved or highly tessellated regions, such as cylindrical surfaces, while avoiding undersampling due to uniform spatial allocation. The approach ensures reproducibility and consistent point cloud sizes while maintaining geometric fidelity.

[1] Curvature is computed as $\kappa(u) = \frac{|x'(u)y''(u) - y'(u)x''(u)|}{(x'(u)^2 + y'(u)^2)^{3/2}}$, with a sampling weight $w(u) = \kappa(u)^{0.5}$ guiding point selection along the B-spline curve $\gamma(u)$.

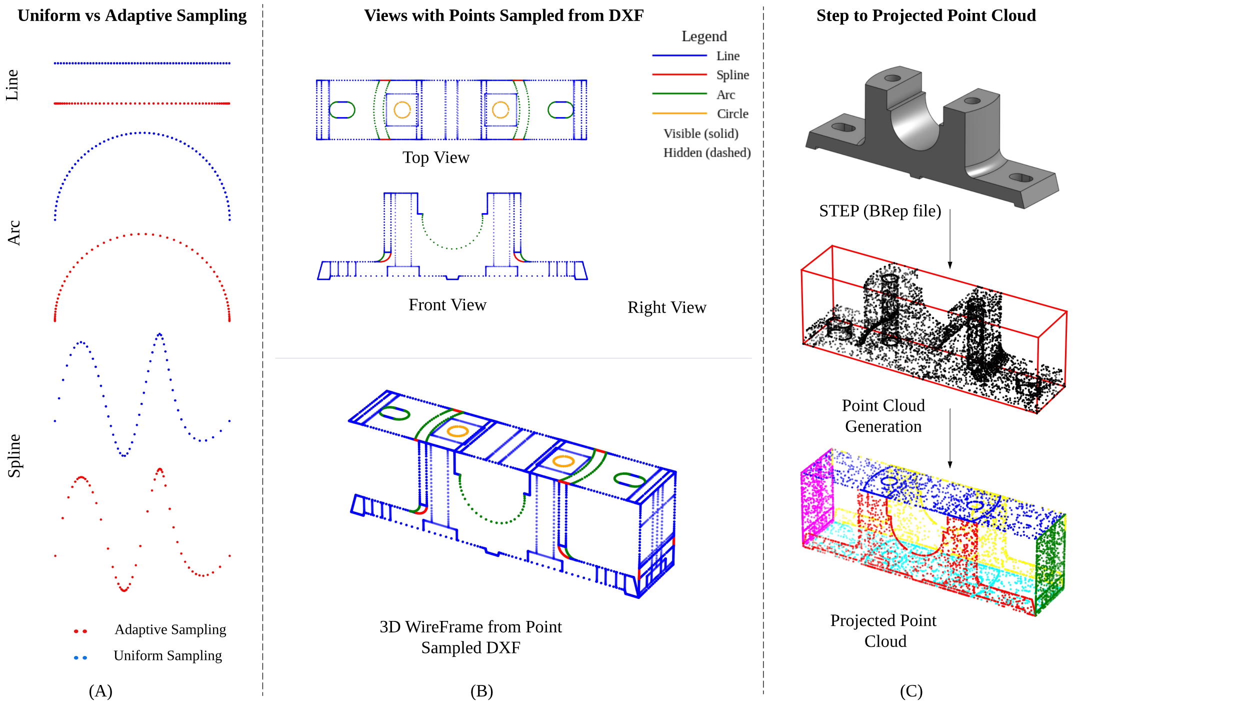


Fig. 5. Fig. A: Uniform vs Adaptive Sampling. Fig. B: Views with Points Sampled from DXF. The figure illustrates the transformation of standard DXF orthographic views into a unified 3D wireframe through adaptive sampling. Fig. C: STEP to Point Cloud. The CAD model is converted into a surface-sampled point cloud and projected for alignment in a shared embedding space.

To generate the projected point cloud, we first compute an oriented bounding box (OBB) for each CAD model. Unlike an axis-aligned bounding box, the OBB aligns with the object's local coordinate system, preserving its intrinsic orientation. The minimum and maximum extents along the local axes define the bounding volume. Each sampled surface point is then orthogonally projected onto the six faces of the OBB. This results in a structured 3D point cloud that captures per-face projections of the object's geometry. Fig. 5 C illustrates this conversion pipeline from STEP files to projected point clouds.

The OBB is first centered by translating it to the origin, and then normalized by uniformly scaling it such that its longest dimension equals one. This effectively fits the geometry within a unit cube while preserving the aspect ratio and intrinsic orientation of the projections. While certain axes may not span the full $[-0.5, 0.5]$ range, this normalization ensures consistent and scale-invariant representation across the dataset.

5. Network Architecture

5.1. Query Encoder

We adapted Graphormer [16] as the encoder for 2D orthographic views due to its ability to model global structural relationships. The encoder operates on fully connected graphs constructed from geometric entities extracted via DXF preprocessing (Section 4.2), with node features defined in Eq. 4 and edge attributes encoding spatial proximity.

Similarly to the original Graphormer [16], we incorporate the centrality encoding to capture local structural roles. Each node is augmented with degree-based embeddings derived from its in-degree and out-degree.

5.1.1. Geometric and Visibility-Aware Feature Encoding

To account for hidden entities, we introduce a dedicated visibility encoding mechanism. Each entity is annotated with a 4-dimensional binary visibility vector v_b, representing its occlusion status in orthographic views. This vector is projected into a learned latent space via a linear transformation $\phi_{\text{vis}}(v_b)$ and concatenated with the geometric features, forming part of a visibility-aware encoding pipeline (see Fig. 6) that integrates view, entity, and geometric attributes.

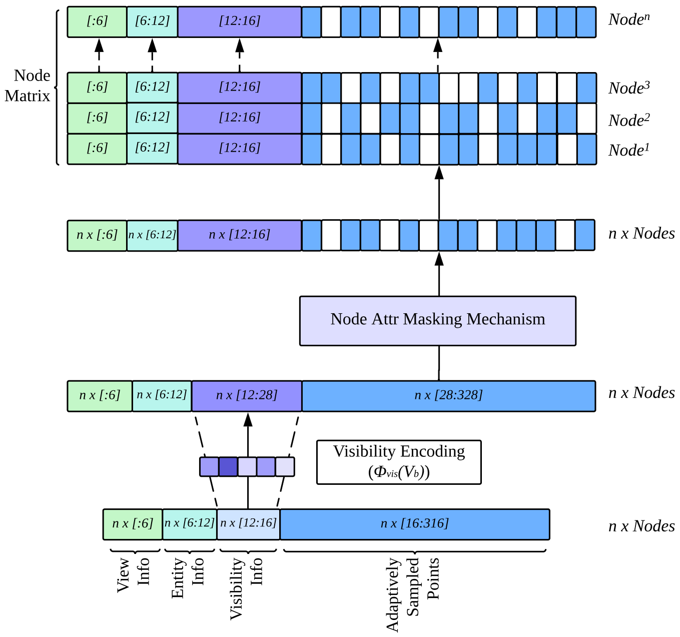


Fig. 6. Node Attribute Processing Pipeline: Encoding View, Entity, Visibility, and Sampled Geometry Features.

To ensure consistent input dimensionality across all geometric entities, we sample 100 points per entity, regardless of complexity. While this guarantees fixed-size input, it can introduce redundancy for simple shapes such as short lines or arcs. To address this redundancy, we employ a feature-masking mechanism that selectively suppresses less informative points while preserving critical geometric details.

We apply selective masking to the last 300 dimensions of each node's feature vector, representing 100 sampled (x, y, z) points. These are reshaped to $\mathbb{R}^{n\times100\times3}$, and attention scores are computed via a learnable projection:

$$\begin{aligned}\alpha &= \mathrm{softmax}(X_{(-300:)}W), \quad \text{where } W \in \mathbb{R}^{3\times1}\\ X_{(-300:)} &= X_{(-300:)} \odot \mathbb{1}[\alpha \geq \mu(\alpha, \dim = 1)]\end{aligned} \tag{6}$$

Only 3D points with attention above the per-node mean are retained, and the masked features are flattened back to $\mathbb{R}^{n\times300}$ for downstream use.

5.1.2. Proximity-Based Spatial Encoding

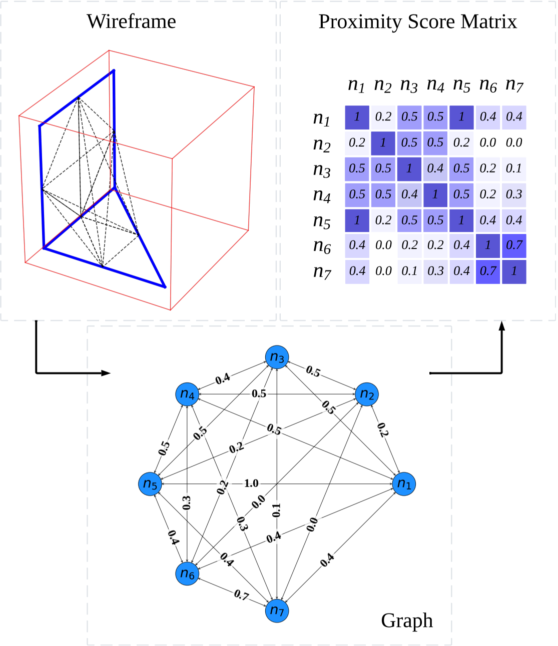
While the original Graphormer architecture models structural relationships using discrete shortest-path distances between nodes, our fully connected graph requires a more geometrically intuitive formulation. We take advantage of the proximity-based spatial encoding defined in Eq. 5, where the notion of proximity is explicitly encoded via edge attributes reflecting normalized Euclidean closeness between entity centers.

This approach ensures that spatially closer entities are assigned higher attention scores during self-attention, promoting local structure preservation while still allowing for global context aggregation.

The spatial bias term b_{ij} is defined as:

$$b_{ij} = \beta \cdot w_{ij} \tag{7}$$

where β is a learnable scalar parameter, and w_{ij} is the proximity-based weight between nodes i and j derived from the normalized Euclidean closeness as defined in Eq. 5.



	n_1	n_2	n_3	n_4	n_5	n_6	n_7
n_1	1	0.2	0.5	0.5	1	0.4	0.4
n_2	0.2	1	0.5	0.5	0.2	0.0	0.0
n_3	0.5	0.5	1	0.4	0.5	0.2	0.1
n_4	0.5	0.5	0.4	1	0.5	0.2	0.3
n_5	1	0.2	0.5	0.5	1	0.4	0.4
n_6	0.4	0.0	0.2	0.2	0.4	1	0.7
n_7	0.4	0.0	0.1	0.3	0.4	0.7	1

Fig. 7. Illustration of proximity-based spatial encoding using sample back and bottom views. Nodes represent geometric entities; edge weights correspond to normalized proximity scores used to compute the bias matrix.

Fig. 7 illustrates this process using back and bottom orthographic views. The parsed geometric entities are converted to fully connected graph nodes, with edge weights derived from proximity scores. The resulting wireframe graph forms a proximity score matrix encoding spatial affinities between node pairs.

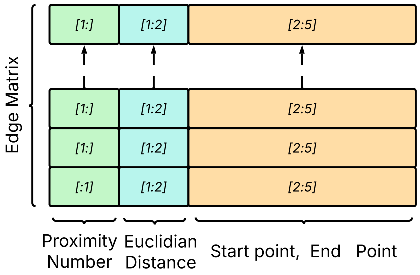


Fig. 8. Edge Attribute Matrix Construction based on Entity Proximity and Spatial Relationships.

5.1.3. Edge Encoding

We adopt a straightforward approach to edge encoding using basic geometric cues, as illustrated in Fig. 8. Edge biases c_{ij} are defined based on normalized proximity and relative spatial positions between entity centers, as described in Eq. 5.

The edge-aware bias term c_{ij} is computed simply as:

$$c_{ij} = a_{ij} \cdot w_E \tag{8}$$

where $a_{ij} \in \mathbb{R}^{d_E}$ is the complete edge attribute vector between nodes i and j that encodes proximity and spatial relationships, and $w_E \in \mathbb{R}^{d_E}$ is a learnable weight vector associated with edge features.

5.1.4. Graphormer Layers with Spatial and Edge Bias

The core of the network consists of a stack of L Graphormer encoder layers. Each layer applies multi-head self-attention, enhanced with both spatial and edge-based biases:

$$\mathrm{Attention}(i, j) = \frac{q_i^\top k_j}{\sqrt{d}} + b_{ij} + c_{ij} \tag{9}$$

Where:

- $q_i = W_Q h_i$ is the query vector of node i,
- $k_j = W_K h_j$ is the key vector of node j,
- d is the dimensionality of the query and key vectors.

Here, W_Q and $W_K \in \mathbb{R}^{d\times d}$ are learned projection matrices, and $h_i, h_j \in \mathbb{R}^d$ are the input feature embeddings of nodes i and j, respectively. The term b_{ij} denotes the spatial bias (Eq. 7), while c_{ij} denotes the edge-aware bias (Eq. 8). Each attention block is followed by a position-wise feedforward network with residual connections and layer normalization, consistent with the original Transformer architecture.

5.1.5. Graph-Level Representation and Pooling

To obtain a global graph embedding $h_G \in \mathbb{R}^{d_{\text{out}}}$, we apply the mean pooling to the node representations in the final Graphormer layer:

$$h_G = \frac{1}{|N|}\sum_{n\in N} h_n \tag{10}$$

where h_n denotes the final hidden representation of node n, and $|N|$ is the total number of nodes in the graph.

5.2. Label Encoder

To encode 3D CAD models, we employ PointNet [17] on projected point clouds (Section 4.3), producing shape-aware embeddings. PointNet is chosen for its permutation invariance, robustness to noise, and computational efficiency compared to volumetric or transformer-based alternatives. In our

Table 2. Training implementation details and encoder configurations for the dual-encoder contrastive learning framework.

Parameter	Query Encoder	Label Encoder
Backbone	Graphormer [16]	PointNet [17]
#Layers	4	3 MLP blocks
Hidden Dim d	256	1024 (pre-proj)
FFN Dim	512	512→256 (MLP)
#Attention Heads	8	–
Head Dim	32	–
FFN Dropout	0.1	–
Attention Dropout	0.1	–
Pooling Type	Mean Pooling	Global Max
Output Embedding Dim	256	256
Feature Transform	–	Enabled
Shared Contrastive Learning Settings		
Optimizer	AdamW (β_1=0.9, β_2=0.999)	
Learning Rate	1e-4	
Weight Decay	1e-5	
Scheduler	Cosine Annealing (Tmax = 100)	
Batch Size	32	
Epochs	100	
Contrastive Loss Temp. (τ)	0.07	
Feature Trans. Reg. Weight	0.001 (Pointnet only)	
Implementation Framework	PyTorch Lightning 2.0	
Hardware used	RTX 4080, Intel i7-12700	

ablation study (Section 9), it outperformed Point Cloud Transformer [37] in both retrieval accuracy and training stability.

The resulting 1024-dimensional global descriptor is passed through a two-layer MLP comprising linear layers with 512 and d_{out} units, batch normalization, and ReLU activations. This yields the final embedding $h_P \in \mathbb{R}^{d_{\text{out}}}$, aligned with the query encoder's latent space.

6. Training Methodology, Loss Function, and Implementation Details

Our framework uses the dual encoder setup in Section 5, where DXF wireframes and 3D point clouds are mapped to a shared 256-dimensional space. The overall training procedure, including feature extraction, similarity computation, and contrastive optimization, is summarized in Algorithm 1. Training implementation details and model configurations for the query encoder (Graphormer) and label encoder (PointNet) are provided in Table 2. The dataset of ∼283,000 filtered CAD models is randomly split into 70% for training (∼198,100 samples), 15% for validation (∼42,450 samples), and 15% for testing (∼42,450 samples).

6.1. Contrastive Learning Objective

Both encoders are trained jointly using an InfoNCE contrastive loss formulation [38]:

$$\mathcal{L}_{\text{contrastive}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp\left(s(e_g^i, e_p^i)/\tau\right)}{\sum_{j=1}^{N}\exp\left(s(e_g^i, e_p^j)/\tau\right)} \tag{11}$$

where $s(\cdot,\cdot)$ denotes cosine similarity, e_g^i and e_p^i are the embeddings of the i^{th} sample from the graph and projection encoders respectively, τ is a temperature parameter controlling the distribution sharpness, and N is the batch size.

Algorithm 1: Training Strategy for Cross-Modal Retrieval

```
Input: Paired dataset $\mathcal{D} = \{(G, P)\}$; learning rate $\eta$;
       temperature $\tau$; regularization weight $\lambda$
Output: Trained encoders $f_G$ and $f_P$
Initialize encoders $f_G$, $f_P$ with parameters $\theta_G$, $\theta_P$
foreach epoch do
    foreach batch $\{(G_b, P_b)\}$ do
        $\text{Attn}(i, j) \leftarrow \frac{\mathbf{q}_i^\top \mathbf{k}_j}{\sqrt{d}} + \beta w_{ij} + \mathbf{a}_{ij} \cdot \mathbf{w}_E$ // Node attention
        $z_G \leftarrow f_G(G_b)$ // Encode DXF graph
        $z_P, T \leftarrow f_P(P_b)$ // Encode point cloud
        $\hat{z}_G \leftarrow z_G/\|z_G\|, \quad \hat{z}_P \leftarrow z_P/\|z_P\|$ // Normalize
        $L_{ij} \leftarrow \hat{z}_G^i \cdot \hat{z}_P^j/\tau$ // Similarity logits
        $L_{\text{cls}} \leftarrow \text{CrossEntropy}(L, \text{targets})$ // Contrastive loss
        $L_{\text{reg}} \leftarrow \sum \|TT^\top - I\|_F^2$ // Transform regularizer
        $L_{\text{total}} \leftarrow L_{\text{cls}} + \lambda \cdot L_{\text{reg}}$ // Total loss
        $\theta_G \leftarrow \theta_G - \eta \nabla_{\theta_G} L_{\text{total}}$
        $\theta_P \leftarrow \theta_P - \eta \nabla_{\theta_P} L_{\text{total}}$
```

To further stabilize training and promote invariance in geometric transformations, we incorporate a regularization term on the feature transformation matrix, following the technique introduced in PointNet [17]:

$$\mathcal{L}_{\text{reg}} = \left\| I - AA^\top \right\|_F^2 \tag{12}$$

where A is the learned transformation matrix and $\|\cdot\|_F$ denotes the Frobenius norm. This term encourages A being close to orthogonal, thus preserving the geometric structure of point cloud features and preventing degenerate mappings.

The final loss function combines these components with a weighting factor:

$$\mathcal{L} = \mathcal{L}_{\text{contrastive}} + \lambda \mathcal{L}_{\text{reg}} \tag{13}$$

We adopt $\lambda = 0.001$ in alignment with the empirically validated configuration proposed in PointNet [17], where it was shown to improve training stability and generalization.

7. Experimental Results and Analysis

7.1. Experimental Setup

We trained two categories of models using consistent hyperparameter settings to ensure fair comparison, with the exception of Model M6, which used half the batch size due to computational constraints. As the OrthoCAD-322K dataset does not contain class labels, retrieval performance was evaluated using three standard metrics: Top-1 accuracy, Top-5 accuracy, and Mean Reciprocal Rank (MRR). All experiments were conducted on the filtered subset of ∼283K models from OrthoCAD-322K.

- Fixed view models (M1–M6): Each model was trained with a fixed number of orthographic views, incrementally added in canonical order: front, top, right, left, back, and bottom. For example, the 1 view (M1) setting corresponds only to the front view, while the 6 view (M6) setting includes all six standard projections.

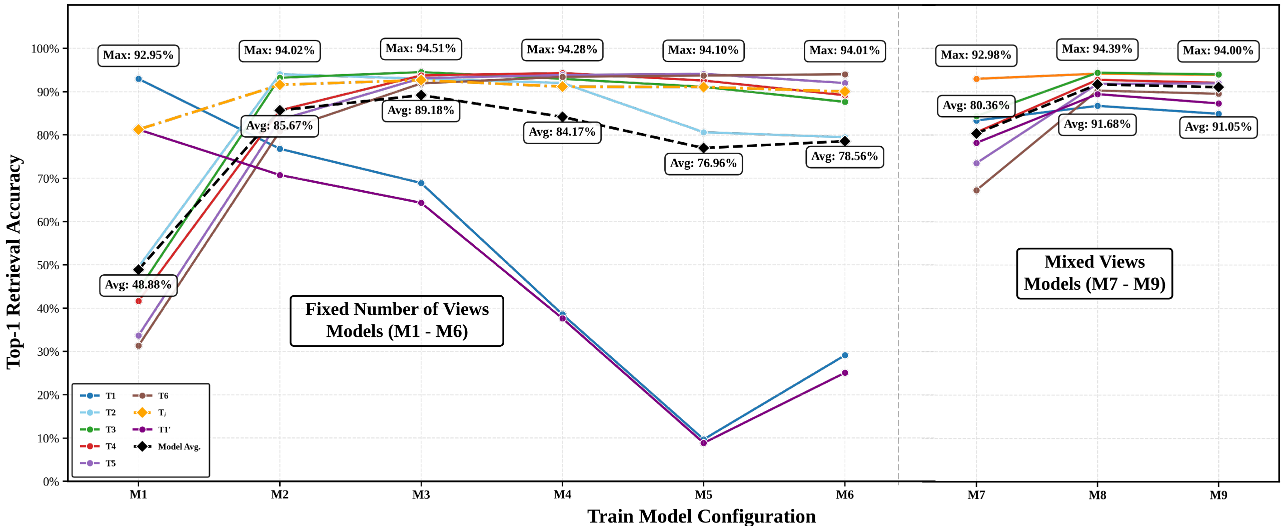


Fig. 9. Top-1 retrieval accuracy under varying orthographic view configurations. Left: Fixed-view models (M1–M6) show reduced accuracy with limited views (e.g., M1). Right: Mixed-view models (M7–M9) demonstrate greater robustness. For example M7, trained with 50% one-view and 50% two-view samples. Training with ≥3 views (e.g., M3–M4, M8) consistently yields higher max and average accuracy across test sets (T1–T6).

Table 3. Top-1 retrieval accuracy (%) across models (M1–M9) and test sets (T1–T6 and T1′). Columns T1′ and T'_i report the mean accuracy (μ) with standard deviation (σ). The T'_i column corresponds to the randomized test set used for M1–M6 (e.g., T2′ for M2), and the best randomized result for M7–M9.

M T	T1	T1′ ($\mu \pm \sigma$)	T2	T3	T4	T5	T6	T'_i ($\mu \pm \sigma$)
M1	92.95	79.45 ± 2.06	49.48	44.23	41.63	33.70	31.32	79.45 ± 2.06
M2	76.80	69.34 ± 1.24	94.02	93.21	85.66	83.49	80.83	90.70 ± 0.71
M3	68.89	61.61 ± 1.92	92.97	94.51	93.74	93.10	91.89	91.89 ± 0.55
M4	38.57	34.04 ± 2.32	92.00	92.96	94.28	93.83	93.40	89.93 ± 0.67
M5	9.71	5.37 ± 2.43	80.62	91.09	92.56	94.10	93.71	89.83 ± 0.72
M6	29.13	23.61 ± 1.12	79.47	87.63	89.17	91.98	94.01	89.68 ± 0.63
M7	92.96	83.10 ± 1.18	93.52	91.86	79.36	72.67	66.78	91.34 ± 0.66
M8	93.17	84.52 ± 1.09	94.05	94.05	93.32	91.75	90.07	91.51 ± 0.60
M9	92.31	83.09 ± 1.11	94.02	94.07	94.10	92.00	89.34	91.07 ± 0.58

- M1’: A variant of M1 trained on single orthographic views, where the single view is randomly selected from any of the six standard directions: front, right, back, left, top, or bottom.
- Mixed view models (M7–M9): Each model was trained on a balanced dataset comprising equal proportions of samples with different numbers of orthographic views:
  - M7: equally sampled from 1 and 2 views
  - M8: equally sampled from 1, 2, and 3 views
  - M9: equally sampled from 1, 2, 3, and 4 views
- Fixed view test sets (T1–T6): Contain 1 to 6 orthographic views in a predefined order, matching the view counts used during training. *(e.g., T3 = front, top, right).*
- Random-view test sets (T1’–T6’) also contain 1 to 6 views, but the views are selected in random order for each sample (e.g., a T3’ sample might include the right, bottom, and back views). We generated five randomized replicates for each of the six randomized view sets (T1’–T6’), resulting in 5 × 6 diverse subsets.

We evaluated cross-generalization performance by testing across all model-test set combinations (M1–M9 × T1–T6). For the random-view sets (M1–M9 × T1’–T6’), we repeated testing five times and report the mean ± standard deviation in Table 3.

7.2. Critical Analysis of Cross-Modal Generalization Results

Table 3 reports the Top-1 retrieval accuracy across all combinations of models and test sets, with performance trends visualized in Fig. 9. Table 4 presents the Top-5 accuracy and Mean Reciprocal Rank (MRR) for the best-performing configurations of models M1 through M6. Our cross-modal evaluation reveals the following key findings:

Table 4. Top-5 retrieval accuracy (%) and Mean Reciprocal Rank (MRR) of models M1–M6, each evaluated on its best-performing input setting T_i (e.g., M1 on T1, M2 on T2, etc.).

Metric	M1	M2	M3	M4	M5	M6
Accuracy (T_i)	95.73	96.11	96.71	96.14	95.87	95.06
MRR (T_i)	0.9482	0.9537	0.9567	0.9533	0.9501	0.9426

- *M3 - View Completeness*: Model M3 achieved the highest Top-1 accuracy (94.51%) on fixed three-view inputs (T3), which include the front view and maintained strong performance on randomized views ((91.89 ± 0.55)% on T3’). It also led in Top-5 accuracy (96.71%) and MRR (0.9567) on T3, highlighting consistent top-ranked retrievals. This suggests that three orthographic views represent an optimal balance between geometric completeness and computational efficiency. Given the inherent viewpoint symmetry in orthographic projections, adding more views (M5-M6) beyond three tends to introduce redundant information without yielding significant performance gains.
- *Mixed-View Robustness:* Mixed-view Models (M7-M9), trained with a variable number of input views, exhibited strong generalization compared to fixed view models (M1–M6). Even when evaluated on the randomized single view set (T1’), mixed-view models (M7-M9) consistently outperformed all fixed-view counterparts (M1-M6), demonstrating superior generalization and robustness to varying input configurations.
- *Stability under View Variations:* Across five replicates for each of the six test sets (T1′–T6′), models M2–M9 exhibited a mean Top-1 accuracy drop of roughly 2%–4%, indicating moderate sensitivity to input variation (see last column of Table 3). The low standard deviation (~0.5%–0.7%) across models M2–M9 further confirms consistent generalization under varying view configurations. M1, in contrast, shows a higher standard deviation of 2.06% on T1′.
- *Bias in M1:* Model M1, trained exclusively on front views,

achieved 92.95% on T1 but dropped to (79.45 ± 2.06)% on T1', indicating overfitting to the more detailed yet depth-limited front view. In contrast, Model M1', trained on single views uniformly sampled from all standard directions (e.g., front, top, right), achieved a comparable score of (79.34 ± 0.3)% on T1'. This suggests that M1's strong performance may be driven by dataset bias favoring visual richness of the front view, rather than true view-invariant feature learning.

- *View-Specific Gaps:* This limitation persists beyond M1. As shown in the T1 column of Table 3, models trained with multiple views (M2–M6) exhibit reduced retrieval accuracy on front-view inputs. This gap likely arises from their predominant exposure to depth-rich projections during training, while front views inherently lack explicit depth cues.
- *Benefits of Mixed-View Training:* The view-specific bias observed in M1 and M2–M6, stemming from exclusive exposure to either views lacking depth cues or depth-rich views, was effectively addressed in Models M7–M9. Trained on mixed views, these models exhibited notable improvements in generalization. This underscores the importance of varied viewpoint exposure in promoting view-invariant structural learning, a crucial step toward practical robustness.
- *Limits of One-View Input:* Despite overall improvements, Models M7–M9 trained with varying numbers of views still exhibited a residual drop in accuracy on T1'. This underscores the inherent challenge of recovering 3D structure from single-view inputs that lack depth cues.

7.3. Qualitative Analysis of Retrieved Models

The qualitative results in Fig. 10 are generated using Model M3, evaluated on the test sets T1 through T4. The figure illustrates the effect of varying the number of query orthographic views (V) on CAD retrieval. As V increases, retrieval accuracy improves. With a single view ($V = 1$), retrievals are often imprecise, returning structurally similar but incorrect models. Increasing to $V = 2$ and $V = 3$ leads to progressively more accurate retrievals. Fig. 10 (highlighted within the dotted boundary) also presents challenging cases, such as spring-like shapes and thin parts with multiple holes, where retrieval struggles or fails even with multiple views. These examples underscore the limitations of orthographic projections in capturing fine geometric details and occluded features.

7.4. Robustness to Partial Geometry, Geometric Perturbations, and Viewpoint Deviations

To simulate scenarios with incomplete or corrupted vector files, where some entities may be missing during parsing, we evaluated the robustness of Model M3 by randomly removing a fixed percentage of geometric entities from the query views. The experiment was conducted at drop rates ranging from 10% to 50%, using the T3 test set. As shown in Table 5 A, Model M3 performed well even under these conditions, with accuracy remaining high at 91.89% at a 30% drop rate and 86.96% at 50%, demonstrating its robustness to missing data.

Furthermore, to simulate minor geometric perturbations that may be commonly introduced during design iterations, we augmented the 3D models by adding one or two small protrusions, such as cubes or cylinders, and used these modified versions as retrieval targets. As indicated in Table 5 B, while M3 exhibits robustness to small-scale deformations, its accuracy decreases with larger protrusions, underscoring sensitivity to topological inconsistencies.

To evaluate generalization under consistent viewpoint deviations, we re-assessed model M3 on 10,000 CAD models by applying rigid-body rotations to the entire object along the X-axis at increments of 5° up to 45°, while keeping the retrieval targets unchanged. For each rotation, new orthographic view triplets (Front, Top, Side) were rendered while preserving mutual orthogonality. While Top-1 accuracy showed a marked decline with increasing rotation angles due to geometric distortions in the DXF projections, Top-5 accuracy exhibited a more gradual decrease, maintaining 64.79% at 45°. This suggests that the model retains a degree of retrieval consistency under systematic viewpoint shifts (Table 5 C).

Table 5. Top-1 retrieval accuracy of Model M3 under three robustness scenarios: (i) partial geometry degradation due to entity drop, (ii) structural mismatch induced by synthetic protrusions, and (iii) viewpoint deviations via rotation.

A. Partial Geometry (Entity Drop)						
Entity Drop (%)	**Baseline**	**10**	**20**	**30**	**40**	**50**
Top-1 Accuracy (%)	94.51	94.22	94.10	91.89	89.60	86.96
B. Structural Mismatch (Protrusion-Based)						
Protrusion type	**5%**	**7.5%**	**10%**	**15%**	**2 x 3%**	**2 x 5%**
Top-1 Accuracy (%)	88.41	82.41	79.43	64.76	81.53	74.76
C. Viewpoint Deviation (Global Object Rotation Along X-Axis)						
Rotation Angle (°)	**0**	**5**	**10**	**20**	**30**	**45**
Top-1 Accuracy (%)	91.75	80.37	68.75	49.47	40.99	36.44
Top-5 Accuracy (%)	97.98	96.96	94.47	83.08	69.97	64.79

8. Comparison Study: Encoder Architectures for DXF-Based Retrieval

To benchmark our approach, all state-of-the-art models were trained and evaluated on our proposed dataset across modalities, except for ViT- and MVCNN-based architectures, which were fine-tuned on the proposed dataset using open-source pre-trained weights (see Table 6). In the 3D domain, PointNet outperformed Point Cloud Transformer (PCT) [37], which, despite using global self-attention, saw accuracy drops (4.34% on test set T3 (front, top and right), 6.75% on average) due to its lack of spatial priors. PointNet's architecture, attuned to spatial regularities, proved more effective for structured CAD data.

For 2D DXF vector inputs, while GCN and GAT could encode relational structures, they lacked positional encoding. GraphGPS [39] improved performance by combining local message passing with global attention (91.68% on T3), but the Graphormer variant surpassed all by capturing both spatial proximity and structural context crucial for engineering views.

Raster-based models like MVCNN [40, 41] and Multi-ViT [42] underperformed, with the best configuration (Multi-ViT+Attn) reaching only 88.12% on T3, compared to Graphormer's 94.51%. This performance gap arises because rasterization often obscures fine geometric and topological details, whereas DXF representations preserve vector-level precision critical for engineering-grade retrieval.

We implemented a cross-modal baseline using the Siamese architecture from CADSketchNet [27], which aligns 2D raster sketches with rendered CAD views in a shared embedding space. Although effective in bridging modalities, it achieved

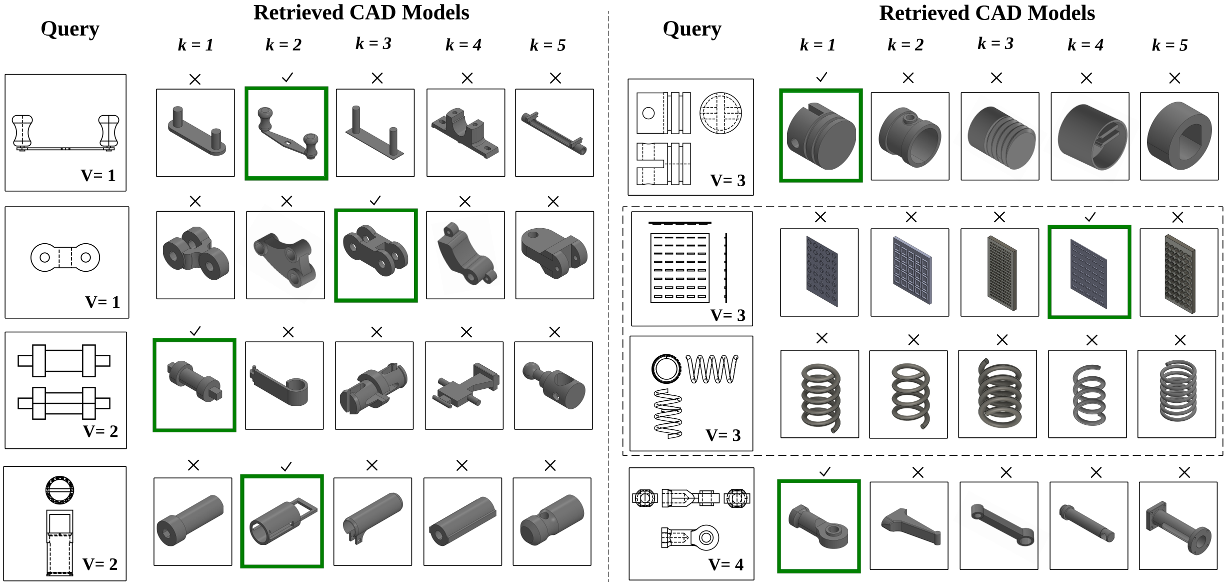


Fig. 10. Qualitative results of CAD model retrieval for orthographic view queries (V = number of query views), evaluated using model M3.

Table 6. Comparison of T3 accuracy/loss and MRR, with average accuracy/loss across views T1–T6. The fixed-view test set T3 consists of the front, top, and right views. Bold values indicate the best-performing model.

Model	T3 Acc (%)	T3 Loss	MRR	Avg T1–T6 Acc (%)	Avg T1–T6 Loss
Ours	**94.51**	**2.7009**	**0.9567**	**89.18**	**3.1673**
GraphGPS	91.68↓	3.2341↑	0.9548↓	85.89↓	3.9825↑
GAT	91.03↓	3.6661↑	0.9544↓	85.59↓	4.3981↑
PCT	90.17↓	3.0818↑	0.9345↓	82.43↓	3.3937↑
GCN	85.18↓	3.0752↑	0.9051↓	68.09↓	3.5285↑
Multi-ViT+Attn	88.12↓	3.5421↑	0.9416↓	80.33↓	4.3287↑
MVCNN-SA	86.67↓	3.8825↑	0.9102↓	76.29↓	4.1001↑
Multi-ViT	85.43↓	5.9808↑	0.8997↓	76.43↓	5.9430↑
MVCNN	84.03↓	3.9492↑	0.8999↓	74.32↓	4.4021↑
CADSketchNet	58.93↓	–	0.8120↓	48.01↓	–

lower accuracy (58.93% on T3), indicating that raster-only inputs may lack the geometric precision needed for CAD retrieval. Loss values are omitted from Table 6 due to differing loss formulations.

Overall, these results highlight that using vector-based encoders with spatial and structural information is key to achieving accurate CAD model retrieval.

9. Ablation Study

We conduct an ablation study to evaluate the contribution of key architectural components in a model trained with three orthographic views (M3). We evaluate performance on T3 and T1–T6, with Table 7 showing accuracy and loss drops for each ablation. Our results highlight several key findings:

- Visibility Encoding and Node Masking: Removing either of these components results in accuracy drops of up to 1.43% (average across T1–T6), with a larger degradation of 2.67% when both are removed, confirming their importance.
- Raw Point Cloud Representation: Bypassing projections causes major performance loss, reflecting the challenge of aligning unordered 3D points with structured 2D graphs. Structured projections retain geometric priors crucial for cross-modal correspondence.
- Loss Function: Replacing contrastive loss with triplet loss results in the largest accuracy drop of 8.65% on T3 and 10.99% on average, emphasizing the effectiveness of contrastive learning for aligning 2D and 3D embeddings.

Table 7. Ablation Study: Change in Top-1 retrieval accuracy (%) and loss of the M3 model on the 3-view test set (T3) and the average across T1–T6.

Configuration	Δ Acc	Δ Avg. Acc	Δ Loss	Δ Avg. Loss
No Visibility Encoding	0.92%↓	1.23%↓	0.5860↑	0.9370↑
Without Node Masking	0.83%↓	1.43%↓	0.2022↑	0.4659↑
No Visibility Encoding+ Without Node Masking	1.98%↓	2.67%↓	1.7299↑	2.0922↑
Raw Point Cloud (No Projection)	7.01%↓	15.01%↓	1.7356↑	2.6224↑
Triplet Loss (Instead of Contrastive)	8.65%↓	10.99%↓	2.5914↑	3.0111↑

10. Conclusion and Future Work

We proposed a cross-modal retrieval framework to retrieve 3D CAD models from 2D orthographic DXF views, addressing the challenge of legacy data reuse in engineering design. Key contributions include a dual encoder architecture that combines Graphormer and PointNet, a novel proximity-based spatial encoding strategy, and the newly developed OrthoCAD-322K dataset. Our method achieves a top-1 retrieval accuracy of 94.51% on fixed three-view inputs (T3), which include the front view and demonstrates strong generalization under incomplete and variable input conditions. Extensive ablation and comparison studies support the effectiveness of our architectural choices. All experiments were conducted on the filtered subset of ~283K models from OrthoCAD-322K. Future work will focus on supporting assembly retrieval, incorporating drawing annotations, and adapting the framework to specialized industries such as aerospace and automotive involving sectional and auxiliary views.

Author Note

During the preparation of this work the author(s) used ChatGPT in order to paraphrase certain sections for improved clarity and readability. After using this tool, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

References

[1] Ganeshram, R, Mills, J. Design intent in 2D CAD: Definition and survey. Computer-aided Design - CAD 2006;3. doi:10.1080/16864360.2006.10738463.

[2] Zhang, C, Pinquié, R, Polette, A, Carasi, G, Charnace, H, Pernot, JP. Automatic 3D CAD models reconstruction from 2D orthographic drawings. Computers Graphics 2023;114. doi:10.1016/j.cag.2023.05.021.

[3] Wang, X, Zheng, J, Hu, Y, Zhu, H, Yu, Q, Zhou, Z. From 2D CAD drawings to 3D parametric models: A vision-language approach 2024;doi:10.48550/arXiv.2412.11892.

[4] Hu, W, Zheng, J, Zhang, Z, Yuan, X, Yin, J, Zhou, Z. PlankAssembly: Robust 3D Reconstruction from Three Orthographic Views with Learnt Shape Programs. In: ICCV. 2023,.

[5] Zhang, C, Polette, A, Pinquié, R, Iida, M, Charnace, H, Pernot, JP. Reinforcement Learning-Based Parametric CAD Models Reconstruction from 2D Orthographic Drawings. 2025. doi:10.2139/ssrn.5174280.

[6] Harish, AB, Prasad, AR. Automated 3D solid reconstruction from 2D CAD using OpenCV. CoRR 2021;abs/2101.04248. URL: https://arxiv.org/abs/2101.04248. arXiv:2101.04248.

[7] Han, W, Xiang, S, Liu, C, Wang, R, Feng, C. SPARE3D: A dataset for SPAtial REasoning on three-view line drawings. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2020,.

[8] Yavartanoo, M, Hong, S, Neshatavar, R, Lee, KM. Text2CAD: Text to 3D CAD generation via technical drawings. 2024. doi:10.48550/arXiv.2411.06206.

[9] Feist, S, Jacques de Sousa, L, Sanhudo, L, Martins, J. Automatic reconstruction of 3D models from 2d drawings: A state-of-the-art review. Eng 2024;5:784–800. doi:10.3390/eng5020042.

[10] Dassault Systèmes, . Finite element analysis: An overview. n.d. URL: https://www.3ds.com/store/cad/finite-element-analysis; accessed: 2025-05-12.

[11] Zhou, Z, Li, Z, Yu, B, Hu, L, Dong, L, Yang, Z, et al. GaussianCAD: Robust self-supervised CAD reconstruction from three orthographic views using 3D gaussian splatting. SSRN Electronic Journal 2024;doi:10.2139/ssrn.5124922.

[12] Furferi, R, Governi, L, Palai, M, Volpe, Y. 3D model retrieval from mechanical drawings analysis. International Journal of Mechanics 2011;5:91–99.

[13] Haghshenas Gorgani, H, Jahantigh, A, Sadeghi, S. 3D model reconstruction from two orthographic views using fuzzy surface analysis. European Journal of Sustainable Development Research 2019;3. doi:10.29333/ejosdr/5726.

[14] Furferi, R, Governi, L, Matteo, P, Yary, V. From 2d orthographic views to 3D pseudo-wireframe: An automatic procedure. International Journal of Computer Applications 2010;5. doi:10.5120/918-1296.

[15] Carrara, A, Nousias, S, Borrmann, A. Vectorgraphnet: Graph attention networks for accurate segmentation of complex technical drawings. 2024. doi:10.48550/arXiv.2410.01336.

[16] Ying, Z, Cai, T, Luo, S, Zheng, S, Ke, G, He, D, et al. Do transformers really perform bad for graph representation? 2021;34:28877–28888.

[17] Ruizhongtai Qi, C, Su, H, Mo, K, Guibas, L. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation 2016;doi:10.48550/arXiv.1612.00593.

[18] Ning, F, Shi, Y, Tong, X, Cai, M, Xu, W. A review and assessment of 3D CAD model retrieval in machine-part design. International Journal of Computer Integrated Manufacturing 2024;:1–23doi:10.1080/0951192X.2024.2382196.

[19] Mandelli, L, Berretti, S. Cad 3d model classification by graph neural networks: A new approach based on step format. ArXiv 2022;abs/2210.16815.

[20] Zhao, S, Yao, H, Zhang, Y, Wang, Y, Liu, S. View-based 3D object retrieval via multi-modal graph learning. Signal Processing 2015;112:110–118. doi:https://doi.org/10.1016/j.sigpro.2014.09.038; signal Processing and Learning Methods for 3D Semantic Analysis.

[21] qian Zhou, W, Jia, J. A learning framework for shape retrieval based on multilayer perceptrons. Pattern Recognit Lett 2019;117:119–130. URL: https://api.semanticscholar.org/CorpusID:58013650.

[22] Liu, YJ, Luo, X, Joneja, A, Ma, CX, Fu, XL, Song, D. User-adaptive sketch-based 3-d cad model retrieval. IEEE Transactions on Automation Science and Engineering 2013;10(3):783–795. doi:10.1109/TASE.2012.2228481.

[23] Pu, J, Lou, K, Ramani, K. A 2D sketch-based user interface for 3D CAD model retrieval. Computer-aided Design - CAD 2005;2. doi:10.1080/16864360.2005.10738335.

[24] Wang, J, He, Y, Tian, H, Cai, H. Retrieving 3D CAD model by freehand sketches for design reuse. Advanced Engineering Informatics 2008;22(3):385–392. doi:https://doi.org/10.1016/j.aei.2008.04.001; collaborative Design and Manufacturing.

[25] Manda, B, Kendre, PP, Dey, S, Muthuganapathy, R. SketchCleanNet A deep learning approach to the enhancement and correction of query sketches for a 3D CAD model retrieval system. Computers Graphics 2022;107:73–83. doi:https://doi.org/10.1016/j.cag.2022.07.006.

[26] Kosalaraman, KK, Kendre, PP, Manilal, RD, Muthuganapathy, R. SketchCleanGAN: A generative network to enhance and correct query sketches for improving 3D CAD model retrieval systems. Computers Graphics 2024;123:104000. doi:https://doi.org/10.1016/j.cag.2024.104000.

[27] Manda, B, Dhayarkar, S, Mitheran, S, Viekash, V, Muthuganapathy, R. 'CADSketchNet' - an annotated sketch dataset for 3D CAD model retrieval with deep neural networks. Computers Graphics 2021;99:100–113. doi:https://doi.org/10.1016/j.cag.2021.07.001.

[28] Wang, B, Zhou, Y. Doodle to Object: Practical Zero-Shot Sketch-Based 3D Shape Retrieval. Proceedings of the AAAI Conference on Artificial Intelligence 2023;37(2):2474–2482. doi:10.1609/aaai.v37i2.25344.

[29] Su, H, Maji, S, Kalogerakis, E, Learned-Miller, E. Multi-view Convolutional Neural Networks for 3D Shape Recognition . In: 2015 IEEE International Conference on Computer Vision (ICCV). Los Alamitos, CA, USA: IEEE Computer Society; 2015, p. 945–953. doi:10.1109/ICCV.2015.114.

[30] Qi, CR, Su, H, NieBner, M, Dai, A, Yan, M, Guibas, LJ. Volumetric and Multi-view CNNs for Object Classification on 3D Data . In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society; 2016, p. 5648–5656. doi:10.1109/CVPR.2016.609.

[31] Koch, Sea. ABC: A big CAD model dataset for geometric deep learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019, p. 9601–9611.

[32] Wu, R, Xiao, C, Zheng, C. DeepCAD: A Deep Generative Network for Computer-Aided Design Models. 2021. doi:10.48550/arXiv.2105.09492.

[33] Willis, KDD, Pu, Y, Luo, J, Chu, H, Du, T, Lambourne, JG, et al. Fusion 360 Gallery: A Dataset and Environment for Programmatic CAD Construction from Human Design Sequences. ACM Transactions on Graphics (TOG) 2021;40(4).

[34] Willis, KD, Jayaraman, PK, Chu, H, Tian, Y, Li, Y, Grandi, D, et al. JoinABLe: Learning Bottom-up Assembly of Parametric CAD Joints. arXiv preprint arXiv:211112772 2021;.

[35] Zhou, S, Tang, T, Zhou, B. CADParser: a learning approach of sequence modeling for B-Rep CAD. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence. IJCAI '23. ISBN 978-1-956792-03-4; 2023,doi:10.24963/ijcai.2023/200.

[36] Pagani, L, Scott, PJ. Curvature based sampling of curves and surfaces. Computer Aided Geometric Design 2018;59:32–48. doi:10.1016/j.cagd.2017.11.004.

[37] Guo, MH, Cai, J, Liu, ZN, Mu, TJ, Martin, RR, Hu, S. PCT: Point cloud transformer. Computational Visual Media 2020;7:187 – 199. URL: https://api.semanticscholar.org/CorpusID:229297794.

[38] Oord, A, Li, Y, Vinyals, O. Representation learning with contrastive predictive coding. 2018. doi:10.48550/arXiv.1807.03748.

[39] Rampášek, L, Galkin, M, Dwivedi, VP, Luu, AT, Wolf, G, Beaini, D. Recipe for a general, powerful, scalable graph transformer. In: Proceedings of the 36th International Conference on Neural Information Processing Systems. NIPS '22; Red Hook, NY, USA: Curran Associates Inc. ISBN 9781713871088; 2022,.

[40] Su, H, Maji, S, Kalogerakis, E, Learned-Miller, E. Multi-view convolutional neural networks for 3D shape recognition. In: Proceedings of the IEEE international conference on computer vision (ICCV). 2015, p. 945–953.

[41] Shajahan, DA, Nayel, V, Muthuganapathy, R. Roof classification from 3-d lidar point clouds using multiview cnn with self-attention. IEEE Geoscience and Remote Sensing Letters 2020;17(8):1465–1469. doi:10.1109/LGRS.2019.2945886.

[42] Dosovitskiy, A, Beyer, L, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations. 2021,URL: https://openreview.net/forum?id=YicbFdNTTy.