ArchCAD-400k: A Large-Scale CAD drawings Dataset and New Baseline for Panoptic Symbol Spotting

Ruifeng Luo [2,1*], **Zhengjie Liu**[1,5*], **Tianxiao Cheng**[2], **Jie Wang**[2], **Tongjie Wang**[2], **Xingguang Wei**[3,6], **Haomin Wang**[3,7], **YanPeng Li**[2], **Fu Chai**[1,5], **Fei Cheng** [1], **Shenglong Ye** [3], **Wenhai Wang**[3], **Yanting Zhang**[8], **Yu Qiao**[3], **Hongjie Zhang**[3†], **Xianzhong Zhao**[1,4†]

[1]Tongji University, [2]Arcplus East China Architectural Design & Research Institute Co., Ltd., [3]Shanghai AI Laboratory, [4]Shanghai Qi Zhi Institute [5]Shanghai Innovation Institute, [6]University of Science and Technology of China, [7]Shanghai Jiao Tong University, [8]Donghua University

`https://github.com/ArchiAI-LAB/ArchCAD`

Abstract

Recognizing symbols in architectural CAD drawings is critical for various advanced engineering applications. In this paper, we propose a novel CAD data annotation engine that leverages intrinsic attributes from systematically archived CAD drawings to automatically generate high-quality annotations, thus significantly reducing manual labeling efforts. Utilizing this engine, we construct ArchCAD-400k, a large-scale CAD dataset consisting of 413,062 chunks from 5538 standardized drawings, making it over 26 times larger than the largest existing CAD dataset. ArchCAD-400k boasts an extended drawing diversity and broader categories, offering line-grained annotations. Furthermore, we present a new baseline model for panoptic symbol spotting, termed Dual-Pathway Symbol Spotter (DPSS). It incorporates an adaptive fusion module to enhance primitive features with complementary image features, achieving state-of-the-art performance and enhanced robustness. Extensive experiments validate the effectiveness of DPSS, demonstrating the value of ArchCAD-400k and its potential to drive innovation in architectural design and construction.

1 Introduction

For a long time, CAD drawings have served as the universal language of architectural design, enabling seamless communication among designers, engineers, and construction personnel through standardized graphical primitives such as arcs, circles, and polylines. Accurate perception of these primitives in 2D CAD drawings is essential for various downstream applications, including automated drawing review and 3D building information modeling (BIM) [1, 2, 3, 4], as this ability enables the optimization of CAD-based workflows while enhancing efficiency and precision in architectural design and construction processes[5, 6].

Building upon earlier research in symbol spotting [7, 8, 9, 10], pioneering work formally defined the task of panoptic symbol spotting [11] for floor plan CAD drawings and established the benchmark dataset FloorPlanCAD [11, 12, 13, 14, 15, 16]. However, the manual annotation of line-grained labels is a highly time-consuming and labor-intensive process, severely limiting the dataset's scale and diversity. This bottleneck restricts the ability to train models that can generalize across diverse

[*]Equal contribution;

[†]Corresponding Authors: nju.zhanghongjie@gmail.com; x.zhao@tongji.edu.cn.

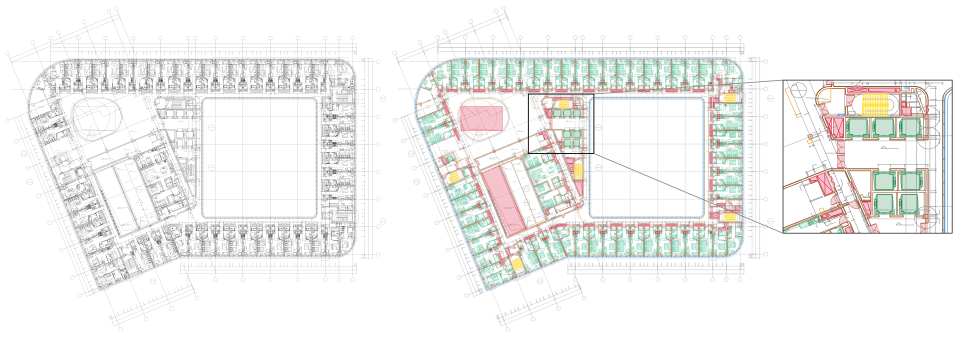

(a) Raw CAD Drawings (b) Panoptic labeled Drawings

Figure 1: An example of annotated drawings in ArchCAD-400k. Each geometric primitive is assigned both semantic and instance labels, with distinct semantic categories represented by different colors and instances visualized through convex hull masks. The chunks are extracted from complete drawings.

building types, varying spatial scales, complex layouts, and a wide range of architectural component categories, thereby hindering their applicability to real-world scenarios.

To address these limitations, we propose an efficient annotation pipeline for line-grained labels, reducing annotation cost from 1,000 person-hours for 16K data to 800 hours for 413K. The core idea is to leverage the structured organization of floor plan drawings, including layers and blocks, to enable scalable and cost-effective large-scale annotation. Layers support bulk labeling via semantic grouping (e.g., doors, windows), while blocks allow instance reuse for repeated elements. As shown in Figure 2, the layer-block structure, inherently designed to enforce drawing standards and ensure consistency in architectural documentation, forms the foundational framework for our automated annotation pipeline. To ensure high data quality, we use completed drawings from top design institutions and a fully vectorized annotation workflow, with expert review of automated outputs.

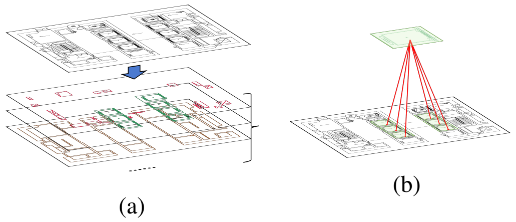

(a) (b)

Figure 2: Layer-Block Structure: (a) Layers organize CAD drawings into functional categories. (b) Blocks define reusable elements for efficiency.

Based on the efficient and accurate data engine, we construct ArchCAD-400k, a large-scale floor plan CAD drawing dataset for the panoptic symbol spotting task. The dataset contains 5,538 complete drawings meticulously selected from 11,917 industry-standard drawings, with a total of 413,062 annotated chunks, surpassing FloorPlanCAD in scale by a factor of 26. While FloorPlanCAD primarily focuses on residential buildings, ArchCAD-400k covers a diverse range of building types, with residential structures accounting for only 14% and large-scale public and commercial facilities forming the majority. The average area of drawings in ArchCAD-400k spans 11,000 m^2, significantly larger than FloorPlanCAD's 1,000 m^2 average, with 51.9% ranging from 1,000 to 10,000 m^2 and 4.4% exceeding 100,000 m^2. Furthermore, ArchCAD-400k introduces a comprehensive semantic categorization, including 27 categories such as structural components (*e.g.*, columns, beams), non-structural elements (*e.g.*, doors, windows), and drawing notations (*e.g.*, axis lines, labels), with 14 categories each containing over 1 million primitives. This extensive scale, diversity, and detailed annotation make ArchCAD-400k a robust resource for advancing AI models in construction industry. An example from ArchCAD-400k is illustrated in Figure 1.

We further propose a novel framework for panoptic symbol spotting, named Dual-Pathway Symbol Spotter (DPSS), which incorporates an adaptive fusion module to effectively enhance primitive features with complementary image features. This design enables the model to achieve superior performance and enhanced robustness compared to existing methods while demonstrating strong scalability and generalization capabilities on larger-scale datasets. In summary, our work presents the following contributions:

(1) We develop a highly efficient annotation pipeline specifically designed for floor plan CAD drawings, which generates high-quality annotations with improved efficiency compared to image-

based manual annotation, significantly reducing the annotation cost from 1,000 person-hours for 16K data to 800 person-hours for 413K data.

(2) We introduce ArchCAD-400k, a large-scale floor plan CAD drawing dataset that surpasses the current largest FloorPlanCAD dataset by an order of magnitude in size and exhibits greater diversity in terms of building types, spatial scales, and architectural component categories.

(3) We propose DPSS, a novel framework for panoptic symbol spotting that achieves state-of-the-art performance on FloorPlanCAD and ArchCAD-400k, surpassing the second-best method by 3% and 10%, respectively, with exceptional accuracy, robustness, and scalability.

2 Related Works

2.1 Floor Plan Datasets

Several datasets have been developed for floor plan analysis. SESYD [9] comprises 1,000 synthetic vectorized documents with ground truth annotations. FPLAN-POLY [10] contains 42 floor plans derived from images [17] for spatial relationship analysis. Cubicasa [3] provides 5,000 floor plan images annotated with over 80 object categories, focusing on residential layouts. RFP [18] includes 2,000 annotated floor plans with detailed room-level information for Asian residential buildings. FloorPlanCAD [11] offers 16,103 vector-graphic floor plans in ".svg" format, annotated across 35 object categories. LS-CAD [16] introduces a test set of 50 full-size CAD drawings with an average area of 1,000 square meters. However, these datasets are limited in scale and rely on labor-intensive annotation, hindering large-scale training.

2.2 Large-scale Vision Datasets

Large-scale vision datasets have significantly advanced deep learning models for complex visual understanding tasks[19, 20, 21, 22, 23, 24, 25, 26, 27], including floor plan recognition. Image-based datasets such as ImageNet [19] (14M images, 21K categories) and COCO [20] (300K images with object detection, segmentation, and captioning labels) have been instrumental in developing state-of-the-art models for object recognition, classification, and scene understanding. These datasets provide rich annotations that enhance model generalization, improving performance across general and domain-specific tasks. Nevertheless, large-scale vector-based datasets and efficient annotation processes for floor plan understanding remain scarce.

2.3 Panoptic Symbol Spotting

The panoptic symbol spotting task, initially proposed in [11], involves the simultaneous detection and classification of architectural symbols (*e.g.*, doors, windows, stairs) in floor plan CAD drawings. While traditional methods [7] focus on countable instances (*e.g.*, windows, tables), Fan *et al.* [11] extended this to uncountable objects (*e.g.*, walls, railings), inspired by [28]. They introduced PanCADNet, which integrates Faster R-CNN [29] for countable instances and Graph Convolutional Networks [30] for uncountable elements. Subsequently, Fan *et al.* [12] proposed CADTransformer, utilizing HRNetV2-W48 [31] and Vision Transformers [32] for primitive tokenization and embedding aggregation. Zheng *et al.* [15] adopted graph-based representations with Graph Attention Networks for semantic and instance-level predictions. Liu *et al.* [14] introduced SymPoint, exploring point set representations, later enhanced by SymPointV2 [13] through layer feature encoding and position-guided training. In contrast, CADSpotting [16] densely samples points along primitives and employs Sliding Window Aggregation for efficient panoptic segmentation of large-scale CAD drawings. While these methods perform well on FloorPlanCAD, they struggle on our more diverse, complex, and large-scale dataset, highlighting the need for robust and scalable solutions.

3 Annotation pipeline of ArchCAD-400k

We carefully gathered over 11917 complete CAD drawings from the industry, covering a wide range of building types. From the initial drawings, 5,538 drawings were validated as strictly conforming to the layer-block organizational standards. Through automated annotation combined with expert refinement, we generated approximately 413,062 chunks with line-grained annotations, each sized

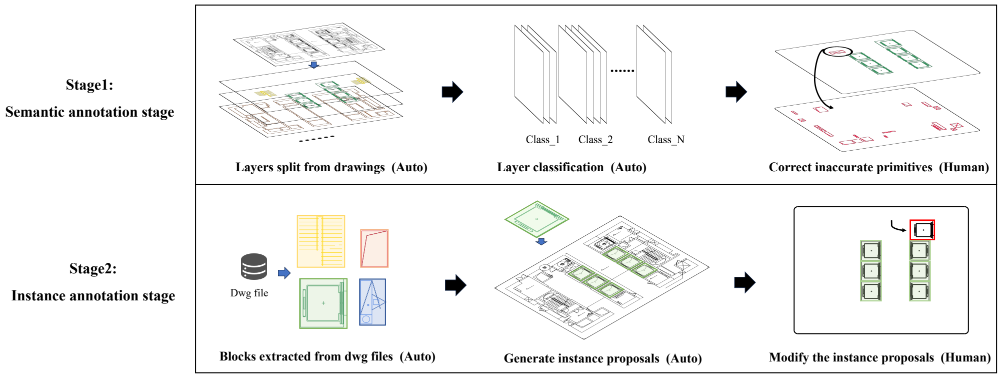

Stage1:
Semantic annotation stage
Layers split from drawings (Auto)
Class_1
Class_2
Class_N
Layer classification (Auto)
Correct inaccurate primitives (Human)
Stage2:
Instance annotation stage
Dwg file
Blocks extracted from dwg files (Auto)
Generate instance proposals (Auto)
Modify the instance proposals (Human)

Figure 3: Overall pipeline of the annotation process. Primitives are labeled by automated methods followed by manual modification.

at 14m×14m, aligning with the design of FloorPlanCAD [11]. The overall annotation pipeline of ArchCAD-400k is shown in Figure 3 and detailed below.

3.1 Annotation Format

The ArchCAD-400k employs a structured annotation format tailored for the panoptic symbol spotting task, as defined in [11]. Each graphical primitive (e.g., line, arc, circle) in the drawing is characterized by a dual-identifier pair, (l_k, z_k), where l_k denotes its semantic category and z_k represents the instance identifier. Primitives sharing the same z_k value are considered to be part of the same instance.

3.2 Layer-Block Standardization Screening

Manual annotation of floor plans is inefficient due to crowded layouts and inconsistent symbols. Standardized drawings use layers for bulk annotation (e.g., door layer) and blocks for identifying repeated instances, clarifying topological and semantic relationships for automation.

However, the feasibility of automated annotation depends on the standardization level of the drawings. Non-standard drawings can have ambiguous layer names and mixed primitives, leading to semantic confusion. Notably, professional design institutes typically maintain internal layering standards, which can be normalized and processed to establish machine-interpretable structures. Thus, we restrict our data to completed drawings from leading design institutions and apply a layer naming validation algorithm against a reference table, discarding drawings with over 5% deviation. This dual quality control ensures standardization of the input data, providing a solid foundation for future automated annotation based on layer-block standardization. The reference table is provided in Appendix A.

3.3 Automated Annotation with Expert Refinement

The automated annotation process based on layer-block standardization greatly reduces manual workload and enables large-scale annotation. However, its accuracy is limited, especially with non-standard drawings that escape standardization. Semantic ambiguities and mixed primitives within layers often cause errors, necessitating human correction. To improve accuracy, we engaged 10 experienced architectural drafters with over 3 years of experience to refine annotations using a vector-based interface. Unlike typical image-based annotation methods such as bounding boxes or polygons, direct vector editing avoids raster-to-vector conversion errors and resolves issues like overlapping instances.

We adopted a two-stage annotation system as shown in Figure 3. In the semantic stage, layers are categorized by names and content, with their semantics displayed in for expert review and correction. In the instance stage, key categories are isolated and instance masks are proposed from block information, then manually refined. We also developed tools to automatically check layer-block compliance, flagging errors for quick human correction. Using this system, we annotated 5,538

Table 1: Comparison of ArchCAD-400k to existing CAD drawing datasets.

Dataset	Source	Scale		Data Format		Annotation		Elements		
		Size	Total Area	Raster	Vector	Type	Method	Architectural	Structural	Notation
FPLAN-POLY [10]	Internet	48	$<5 \times 10^4$ m^2	✓	✗	Instance	Human	✓	✗	✗
SESYD [9]	Synthetic	1000	$<5 \times 10^5$ m^2	✓	✓	Instance	Human	✓	✗	✗
FloorPlanCAD [11]	Industry	16K	1.6×10^6 m^2	✓	✓	Panoptic	Human	✓	✗	✗
ArchCAD-400k	Industry	413K	8.1×10^7 m^2	✓	✓	Panoptic	Auto & Human	✓	✓	✓

drawings in 800 hours, which is over 10 times more efficient than the FloorPlanCAD dataset while maintaining high accuracy and reliability.

Ethical and Copyright Considerations. The dataset used in this work is derived from architectural or design drawings, which may contain potentially sensitive or proprietary information. To ensure compliance with ethical and copyright standards, we apply strict data anonymization before training or release, including removal of identifiable text and irreversible obfuscation of metadata, so that the original content of any individual project cannot be reconstructed. All data is used solely for research under academic fair use, and no raw data that may infringe copyright or IP rights is released. These measures ensure compliance with ethical standards and protection of PII.

4 Exploring ArchCAD-400k

We compare our ArchCAD-400k with existing datasets, as summarized in Table 1. Publicly available datasets, such as SESYD [9], FPLAN-POLY [10], and FloorPlanCAD [11], exhibit limitations in terms of source, scale, data format, annotation type and method, as well as the range of elements included in floor plans. These constraints affect their applicability to real-world architectural analysis.

4.1 Extended Building Diversity

Previous floor plan datasets predominantly originated from residential buildings, with a primary emphasis on indoor layouts. This limitation constrained their applicability in analyzing a wider variety of project types. In contrast, our dataset is curated from a more diverse array of buildings, as illustrated in Figure 4a, with residential structures comprising only 14% of the total. A substantial portion consists of large-scale public and commercial buildings, including office complexes, industrial parks, and other expansive facilities. This diverse composition provides a richer and more representative architectural dataset, enabling applications across a wider variety of building scenarios.

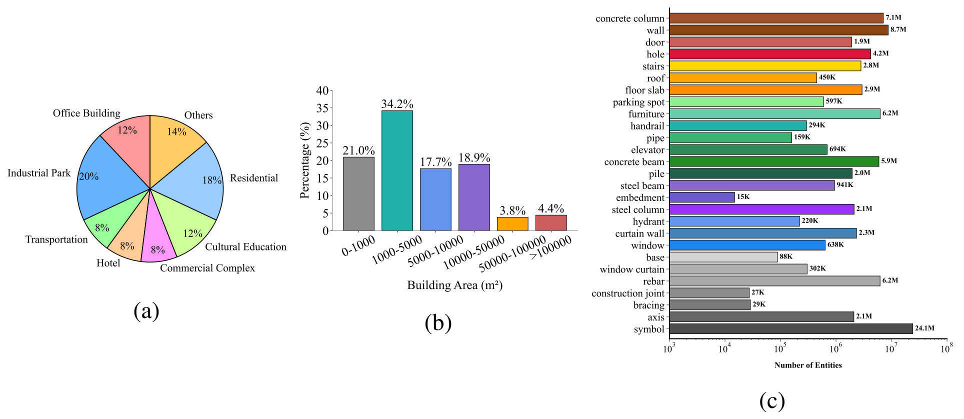


Figure 4: (a) Different project types of drawings; (b) Area distribution of drawings; (c) Number of annotated primitives for 27 semantic classes, where bar colors match each class's visualization.

4.2 Larger Data and Spatial Scale

ArchCAD-400k stands out for its unprecedented scale, comprising 5,538 complete drawings and 413,062 chunks, surpassing FloorPlanCAD's 15,663 chunks by over 26 times. This substantial scale significantly enhances the robustness and generalization capabilities of models for the spotting task.

In addition to the data scale, ArchCAD-400k features much larger drawings compared to existing datasets, which are generally restricted to small areas under 1,000 m^2. The average drawing in our dataset spans 11,000 m^2, closely reflecting real-world architectural dimensions. As shown in Figure 4b, 51.9% of the drawings cover areas between 1,000 and 10,000,m^2, while 4.4% exceed 100,000 m^2, representing a wide spectrum of building areas and configurations.

4.3 Diverse Element Types

Traditional floor plan datasets typically focus on common non-structural elements, such as furniture, equipment, and decorative features. While such categorizations are sufficient for residential interior layouts, they often fail to capture critical components when applied to more diverse building types, particularly in complex architectural or industrial settings.

To address this limitation, our ArchCAD-400k adopts a comprehensive semantic categorization that extends beyond traditional architectural elements. Specifically, we introduce annotations for structural components (e.g., columns, beams, and holes) and drawing notations (e.g., axis lines, labels, and markers) in addition to non-structural elements. Structural components are fundamental to reconstructing the overall building structure as they define the load-bearing framework and spatial organization of the architectural design. In contrast, drawing notations are ubiquitous in real-world drawings and play a critical role in ensuring accurate interpretation and practical implementation. This tripartite categorization, encompassing non-structural, structural, and notation elements, enables a broader range of applications, including indoor navigation, architectural design, and structural analysis.

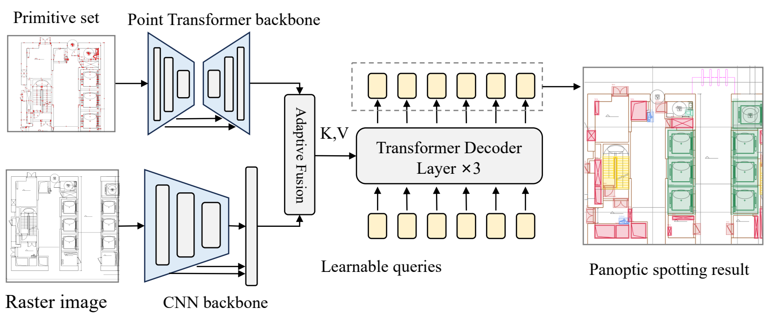
ArchCAD-400k provides detailed annotations for 27 categories, with the distribution of primitives across these categories visually represented in Figure 4c. Notably, 14 of these categories each encompass more than 1 million primitives, underscoring the extensive scale of our dataset. Additionally, our dataset includes 7 countable instance categories. While this count is fewer than that of FloorPlanCAD, our dataset surpasses it in terms of diversity and complexity within each category, presenting a more formidable challenge. For instance, we consolidate various fine-grained furniture types from FloorPlanCAD, such as chair, table, and bed, into a single unified “furniture" category. Visual examples of the categories can be found in Appendix A.

5 Method

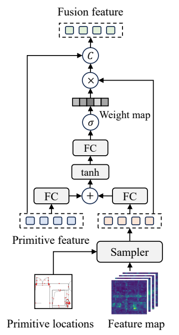
To enhance panoptic symbol spotting on large-scale datasets, we propose **D**ual-**P**athway **S**ymbol **S**potter (DPSS), a robust framework free from prior knowledge like color or layer cues. As shown in Figure 5, DPSS employs a dual-path feature extractor: an image branch encodes rendered graphics for global context, while a point cloud branch captures geometric details by treating primitives as points. An adaptive fusion module aligns these multimodal features, and a transformer decoder performs panoptic segmentation for precise symbol spotting in complex diagrams.

5.1 Two-stream Primitive Encoding Module

The extraction of high-quality features for vector graphic panoramic segmentation is crucial as it directly affects segmentation performance. Unlike raster images, vector graphics consist of primitives, yet research on their modeling remains limited, highlighting the need for better feature extraction methods. CADTransformer [12] rasterizes vector graphics and samples features at the primitive centers, while SymPoint [14] encodes primitives as points using geometric attributes and a point cloud encoder[33]. Image-based approaches capture semantics well but struggle with fine-grained tasks, whereas point-based methods handle instances effectively but are less suited for sparse structures like walls. To address this, we combine visual and geometric encoding. Our visual encoder, based on HRNetV2 [34], extracts feature maps from rendered images. Meanwhile, a Point Transformer-



(a) DPSS Framework



(b) Adaptive fusion

Figure 5: The overall architecture of the proposed Dual-Pathway Symbol Spotter (DPSS). CAD drawings are encoded through both point backbone and image backbone. Features from dual pathways are aligned and fused in adaptive fusion module, then pass through a transformer decoder to generate mask predictions and classifications at primitive level.

inspired point cloud encoder generates geometric features for each primitive, ensuring a more comprehensive representation.

5.2 Adaptive Fusion Module

We use an image sampler to extract semantic features for each primitive. Given the primitive center (x_c, y_c) and the feature map $F \in \mathbb{R}^{H\times W\times C}$, the sampler produces point-wise image features V. Since sampling points may fall between pixels, bilinear interpolation is applied:

$$V_p = \mathcal{K}(F_{N(p')}), \tag{1}$$

where V_p is the extracted feature, $\mathcal{K}$ is the interpolation function, and $F_{N(p')}$ denotes neighboring pixels. The combination of primitive and image features is challenging due to size variations, noise, and overlapping. To address this, we introduce a geometry-guided fusion layer. First, graphic features X_p and image features V_p are transformed via an MLP, then concatenated and pass through an activation function σ. A weight matrix w is computed as:

$$w = \sigma(W_1 \tanh(W_2 X_p + W_3 V_p)), \tag{2}$$

where W_1, W_2, W_3 are learnable parameters. Finally, the weighted features yield the representation:

$$U_p = \text{concat}(X_p, wV_p). \tag{3}$$

5.3 Decoder and Loss Function

The decoder is designed based on the architectures of DETR [35] and Mask2Former [36], achieving panoptic symbol spotting by predicting primitive-level masks along with their categories. The loss function is composed of a standard cross-entropy loss (L_{cls}) for class predictions, a binary cross-entropy loss (L_{bce}), and a Dice loss (L_{dice}) [37] for mask predictions. The overall loss is formulated as a weighted sum of the three losses $L = \lambda_{cls}L_{cls} + \lambda_{bce}L_{bce} + \lambda_{dice}L_{dice}$, where λ_{cls}, λ_{bce}, and λ_{dice} denote the weight for each loss term respectively.

6 Experiments

We split our ArchCAD-400k dataset into training, validation, and test sets using a 7:1:2 ratio, ensuring that each drawing and its corresponding annotations appear in only one split. This results in 289,144 annotated samples for training, 41,306 for validation, and 82,612 for testing. Following the definition of panoptic symbol spotting, we evaluate the model performance using Panoptic Quality (PQ), Segmentation Quality (SQ), and Recognition Quality (RQ). The formulation of these metrics can be found in [11]. To evaluate model performance on ArchCAD-400k, we compare existing methods and our proposed method DPSS, with detailed results presented below.

Table 2: Panoptic symbol spotting results on FloorplanCAD [11] dataset

Method	Additional prior inputs	Total			Thing			Stuff		
		PQ	SQ	RQ	PQ	SQ	RQ	PQ	SQ	RQ
SymPointV2[13]	w/	90.1	96.3	93.6	90.8	96.6	94.0	80.8	90.9	88.9
CADSpotting[16]	w/	88.9	95.6	93.0	89.7	96.2	93.2	80.6	89.7	89.8
DPSS	w/	89.5	96.2	93.1	90.4	96.6	93.5	79.7	91.1	87.5
CADTransformer[12]	w/o	68.9	88.3	73.3	78.5	94.0	83.5	58.6	81.9	71.5
GAT-CADNet[15]	w/o	73.7	91.4	80.7	–	–	–	–	–	–
SymPoint[14]	w/o	83.3	91.4	91.1	84.1	94.7	88.8	48.2	69.5	69.4
SymPointV2[13]	w/o	83.2	91.3	91.1	85.8	92.5	92.7	49.3	70.3	70.1
DPSS	w/o	86.2	93.0	92.6	88.0	94.1	93.5	64.7	83.0	77.9

6.1 Implement Details

We adopt HRNetW48 [31] pretrained on COCO-Stuff [38] as the backbone for the visual branch. The point cloud branch uses the PointTransformerV2 [39] encoder pretrained on ScanNetV2 [40]. The model is trained on 8 NVIDIA A800 GPUs and the optimizer is AdamW. On FloorplanCAD [11] dataset, we set a batch size of 2 per GPU and with a learning rate of 2×10^{-4} and a weight decay of 0.1. The model is trained for 50 epochs. On ArchCAD-400k, we set a batch size of 4 per GPU and train the model for 10 epochs with a learning rate of 2×10^{-4} and the same weight decay. Other baseline methods on ArchCAD-400k are trained for 10 epochs under their default configurations. All models were confirmed to have converged before evaluation to ensure fair performance comparison.

6.2 Quantitative Evaluation

Quantitative comparison on FloorPlanCAD. We evaluate multiple methods on the FloorPlanCAD dataset (Tables 2 and 3), under two scenarios: with and without prior information (such as layers and color). This distinction reflects how CAD drawings encode semantics—primitives in the same layer or with similar colors often share categories.

SymPointV2 improves on SymPointV1 by adding layer encoding, while CADSpotting samples primitives based on color and position. As shown in Table 2, DPSS performs comparably to these methods when priors are present. Without them, however, SymPointV2 suffers a notable drop, whereas DPSS stays robust—achieving 3% higher overall PQ and 15% higher "Stuff" PQ than SymPointV2, demonstrating better generalization. In Table 3, DPSS also clearly outperforms others in Semantic and Instance Spotting without priors, including some classical semantic segmentation and instance detection algorithms.

While our method benefits from prior information, we prioritize performance without it, as this better reflects real-world scenarios where such metadata is often unavailable.

Table 3: Semantic and instance spotting results on FloorPlanCAD

Method	Additional prior inputs	Semantic spotting		Instance spotting		
		F1	wF1	AP50	AP75	mAP
SymPointV2 [13]	w/	89.5	88.3	71.3	60.7	60.1
CADSpotting [16]	w/	93.5	93.9	72.2	69.1	69.0
DPSS	w/	93.1	93.2	74.8	71.0	70.8
DeepLabv3+R101 [41]	w/o	68.8	71.4	–	–	–
DINO [42]	w/o	–	–	64.9	54.9	47.5
CADTransformer [11]	w/o	82.2	80.1	–	–	–
SymPoint [14]	w/o	86.8	85.5	66.3	55.7	52.8
SymPointV2 [13]	w/o	87.0	86.3	66.4	57.7	57.5
DPSS	w/o	92.0	93.2	67.0	61.8	61.5

Quantitative comparison on ArchCAD-400k. We evaluate various methods on our ArchCAD-400k dataset (Table 4), comparing DPSS with CADTransformer, SymPoint, and SymPointV2, all without using layer or color priors, to better reflect real-world conditions. Compared to FloorPlanCAD, ArchCAD-400k is more challenging, leading to overall lower performance. For instance, SymPointV2 drops from 83.2% PQ on FloorPlanCAD to 60.5% on ArchCAD-400k —a decrease of over 22%.

Table 4: Panoptic, semantic, and instance symbol spotting results on ArchCAD-400k

Method	Total			Thing			Stuff			Semantic Spotting		Instance Spotting		
	PQ	SQ	RQ	PQ	SQ	RQ	PQ	SQ	RQ	F1	wF1	AP50	AP75	mAP
CADTransformer[12]	60.0	89.7	66.9	52.5	83.6	62.7	70.1	96.7	72.5	84.1	83.4	—	—	—
SymPoint[14]	47.6	86.1	55.3	51.4	91.9	55.9	39.9	73.9	54.0	76.8	62.0	36.1	30.9	30.8
SymPointV2[13]	60.5	88.0	68.8	62.4	91.7	68.1	52.8	73.7	71.7	69.8	69.3	44.9	40.2	39.7
DPSS	70.6	90.2	78.2	65.6	92.4	70.9	77.6	87.8	88.4	87.8	84.1	45.6	41.1	40.7

DPSS shows clear advantages in scalability and robustness. In semantic spotting, it achieves an F1 of 87.8%, outperforming CADTransformer (84.1%), SymPoint (76.8%), and SymPointV2 (69.8%). In instance spotting, it reaches 40.7% mAP, surpassing SymPointV2 (39.7%) and SymPoint (30.8%). For panoptic symbol spotting, DPSS attains a total PQ of 70.6%, significantly ahead of SymPointV2 (60.5%) and SymPoint (47.6%). Notably, its Stuff PQ reaches 77.6%, 24.8% higher than SymPointV2.

In summary, DPSS consistently outperforms baselines across all tasks, even without prior information. Its strong performance on the more complex ArchCAD-400k highlights its scalability, robustness, and practical applicability to real-world CAD scenarios.

6.3 Ablations

Ablation studies on DPSS. To demonstrate the effectiveness of the proposed DPSS, we conducted ablation studies focusing on different encoding strategies. Specifically, we evaluated the impact of using only the primitive encoder, only the image encoder, and the combination of both, as shown in Table 5 (Lines 1–3). The integration of both the image encoder and the primitive encoder yields a 1.3% improvement in PQ compared to using the primitive encoder alone, and a 2.1% gain over using only the image encoder. These results highlight the complementary nature of the two encoding branches. Furthermore, we investigated the role of the adaptive fusion module by replacing it with a simple concatenation strategy for combining image and primitive features, as shown in Table 5 (Lines 3–4). The results show that incorporating our adaptive fusion leads to a significant 3.2% increase in PQ, underscoring its effectiveness in enhancing feature integration.

Table 5: Ablation experiments on FloorPlanCAD

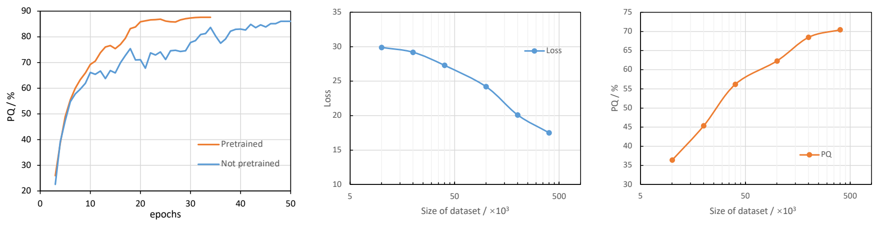
Primitive Encoder	Image Encoder	Adaptive Fusion	PQ	RQ	SQ
✓			81.7	90.1	90.6
	✓		80.9	89.4	90.6
✓	✓		83.0	90.1	92.1
✓	✓	✓	86.2	93.0	92.6

Generalization ability of ArchCAD-400k. To validate the generalization ability of ArchCAD-400k, We implement DPSS on FloorPlanCAD[11] dataset with a dual-pathway encoder pretrained on ArchCAD-400k. Results are illustrated in Figure 6a. The pre-trained model achieves faster and more stable convergence, surpassing the performance of non-pre-trained models at 50 epochs within 30 epochs. It suggests that ArchCAD-400k covers FloorPlanCAD and furthermore has more diverse patterns and properties for models to generalize across datasets.

Scaling performance of ArchCAD-400k. We conducted experiments to validate the scaling law in the context of panoptic symbol spotting. Using a subset of our ArchCAD-400k and adopting DPSS as the baseline network, we evaluated the training performance across different data scales, ranging from 10K samples to the full 400K samples. As demonstrated in Figure 6b and 6c, within 10K to 400K data range, doubling the dataset size consistently reduces the loss by 2.33 and improves the PQ metric by 6.40% in average. The results indicate that, for panoptic symbol spotting tasks, large-scale datasets significantly enhance model performance.

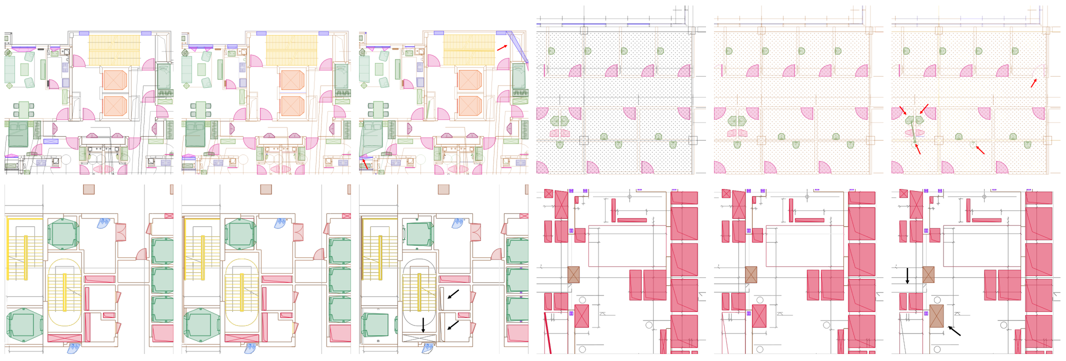
6.4 Qualitative Results

The qualitative results on our ArchCAD-400k are illustrated in Figure 7. In the presented cases, DPSS demonstrates superior performance than SymPointV2 [13]. In architectural diagrams (row 1), characterized by a diversity of component types and numerous interfering lines, DPSS is capable



(a) (b) (c)

Figure 6: (a) Comparison of model performance: pretrained vs. non-pre-trained on ArchCAD-400k; (b) Loss convergence across different dataset sizes; (c) Panoptic quality across different dataset sizes.



(a) Ground truth (b) DPSS (c) SymPointV2 (d) Ground truth (e) DPSS (f) SymPointV2

Figure 7: Qualitative comparison between DPSS and SymPointV2 [13] on FloorPlanCAD(Line1) and ArchCAD-400k (Line2).

of filtering out such disturbances to achieve precise spotting results. In structural drawings (row 2), where the shapes of components exhibit high similarity, DPSS effectively integrates contextual information to discern semantic differences between analogous components.

7 Conclusion

In this work, we address the challenges in panoptic symbol spotting for architectural CAD drawings by introducing an efficient annotation pipeline, a large-scale dataset (ArchCAD-400k), and a novel framework (DPSS). Our pipeline reduces annotation costs, enabling the creation of ArchCAD-400k, which surpasses existing datasets in scale and diversity. With 413,062 annotated chunks from 5,538 drawings, ArchCAD-400k covers diverse building types and spatial scales, advancing AI models in architectural design. DPSS, equipped with an adaptive fusion module to effectively enhance primitive features with complementary image features, achieves state-of-the-art performance on FloorPlanCAD and ArchCAD-400k, demonstrating superior accuracy, robustness, and scalability. We hope that our ArchCAD-400k will catalyze further progress in this domain.

Acknowledgements

This work was supported by the Shanghai Qi Zhi Institute (SQZ202309), the National Natural Science Foundation of China (No. 62206046), and the Shanghai Committee of Science and Technology (No. 22YF1461500).

References

[1] Lukas Kratochvila, Gijs de Jong, Monique Arkesteijn, Simon Bilik, Tomas Zemcik, Karel Horak, and Jan S Rellermeyer. Multi-unit floor plan recognition and reconstruction using improved semantic segmentation of raster-wise floor plans. *arXiv preprint arXiv:2408.01526*, 2024.

[2] Sungsoo Park and Hyeoncheol Kim. 3dplannet: generating 3d models from 2d floor plan images using ensemble methods. *Electronics*, 10(22):2729, 2021.

[3] Ahti Kalervo, Juha Ylioinas, Markus Häikiö, Antti Karhu, and Juho Kannala. Cubicasa5k: A dataset and an improved multi-task model for floorplan image analysis. In *Image Analysis: 21st Scandinavian Conference, SCIA 2019, Norrköping, Sweden, June 11–13, 2019, Proceedings 21*, pages 28–40. Springer, 2019.

[4] Chen Liu, Jiajun Wu, Pushmeet Kohli, and Yasutaka Furukawa. Raster-to-vector: Revisiting floorplan transformation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2195–2203, 2017.

[5] Zhiliang Zeng, Xianzhi Li, Ying Kin Yu, and Chi-Wing Fu. Deep floor plan recognition using a multi-task network with room-boundary-guided attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9096–9104, 2019.

[6] Bingchen Yang, Haiyong Jiang, Hao Pan, and Jun Xiao. Vectorfloorseg: Two-stream graph attention network for vectorized roughcast floorplan segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1358–1367, 2023.

[7] Alireza Rezvanifar, Melissa Cote, and Alexandra Branzan Albu. Symbol spotting for architectural drawings: state-of-the-art and new industry-driven developments. *IPSJ Transactions on Computer Vision and Applications*, 11:1–22, 2019.

[8] Alireza Rezvanifar, Melissa Cote, and Alexandra Branzan Albu. Symbol spotting on digital architectural floor plans using a deep learning-based framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 568–569, 2020.

[9] Mathieu Delalandre, Ernest Valveny, Tony Pridmore, and Dimosthenis Karatzas. Generation of synthetic documents for performance evaluation of symbol recognition & spotting systems. *International Journal on Document Analysis and Recognition (IJDAR)*, 13(3):187–207, 2010.

[10] Marçal Rusiñol, Agnés Borràs, and Josep Lladós. Relational indexing of vectorial primitives for symbol spotting in line-drawing images. *Pattern Recognition Letters*, 31(3):188–201, 2010.

[11] Zhiwen Fan, Lingjie Zhu, Honghua Li, Xiaohao Chen, Siyu Zhu, and Ping Tan. Floorplancad: A large-scale cad drawing dataset for panoptic symbol spotting. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10128–10137, 2021.

[12] Zhiwen Fan, Tianlong Chen, Peihao Wang, and Zhangyang Wang. Cadtransformer: Panoptic symbol spotting transformer for cad drawings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10986–10996, 2022.

[13] Wenlong Liu, Tianyu Yang, Qizhi Yu, and Lei Zhang. Sympoint revolutionized: Boosting panoptic symbol spotting with layer feature enhancement. *arXiv preprint arXiv:2407.01928*, 2024.

[14] Wenlong Liu, Tianyu Yang, Yuhan Wang, Qizhi Yu, and Lei Zhang. Symbol as points: Panoptic symbol spotting via point-based representation. *arXiv preprint arXiv:2401.10556*, 2024.

[15] Zhaohua Zheng, Jianfang Li, Lingjie Zhu, Honghua Li, Frank Petzold, and Ping Tan. Gat-cadnet: Graph attention network for panoptic symbol spotting in cad drawings. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11747–11756, 2022.

[16] Jiazuo Mu, Fuyi Yang, Yanshun Zhang, Junxiong Zhang, Yongjian Luo, Lan Xu, Yujiao Shi, Jingyi Yu, and Yingliang Zhang. Cadspotting: Robust panoptic symbol spotting on large-scale cad drawings. *arXiv preprint arXiv:2412.07377*, 2024.

[17] Xavier Hilaire and Karl Tombre. Robust and accurate vectorization of line drawings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6):890–904, 2006.

[18] Xiaolei Lv, Shengchu Zhao, Xinyang Yu, and Binqiang Zhao. Residential floor plan recognition and reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16717–16726, 2021.

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[21] Hongjie Zhang, Yi Liu, Lu Dong, Yifei Huang, Zhen-Hua Ling, Yali Wang, Limin Wang, and Yu Qiao. Movqa: A benchmark of versatile question-answering for long-form movie understanding. *arXiv preprint arXiv:2312.04817*, 2023.

[22] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *International journal of computer vision*, 128(7):1956–1981, 2020.

[23] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

[24] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016.

[25] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021.

[26] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019.

[27] Yiyi Liao, Jun Xie, and Andreas Geiger. Kitti-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3292–3310, 2022.

[28] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9404–9413, 2019.

[29] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(6):1137–1149, 2016.

[30] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[31] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5693–5703, 2019.

[32] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[33] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021.

[34] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3349–3364, 2020.

[35] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[36] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022.

[37] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. Ieee, 2016.

[38] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff: Thing and stuff classes in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1209–1218, 2018.

[39] Xiaoyang Wu, Yixing Lao, Li Jiang, Xihui Liu, and Hengshuang Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *Advances in Neural Information Processing Systems*, 35:33330–33342, 2022.

[40] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[41] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[42] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In the abstract and introducton, we make the relevent claims on the new large-sacle CAD dataset and new baseline on panoptic symbol spotting as backed by our dataset and experiment sections.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discussed the limitations and potential future work of this project in Appendix C.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: There is no theoretical results associated with this work.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide sufficient details in section 5 to allow reproduction of the main results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Currently, the code and dataset are publicly available at github and huggingface respectively.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: In Section 6.1, we clearly demonstrated various experimental settings, including hyperparameters, model settings, etc.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: Due to the high cost of training, we do not repeat the same experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide detailed descriptions of the compute resources used for our experiments. Details are included in section 6.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We followed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We briefly discuss potential societal impacts in Section 7. The proposed dataset and method can enhance automation and efficiency in CAD-based design workflows by reducing manual effort and improving symbol recognition accuracy. The dataset used in this study has been properly anonymized, with all private or personal information removed. To the best of our knowledge, this research does not pose any explicit negative societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not foresee any high risk for misuse of this work.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All third-party assets (including code, data, and models) used in this work have been properly attributed to their original creators, with full compliance to their respective licenses and terms of use. The dataset is released under the CC-BY-NC 4.0 license, and the code is available under the CC-BY 4.0 license.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The newly released dataset and algorithm are well documented for the open-sourced version, ensuring clarity, reproducibility, and ease of use for the research community.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: Our research does not employ LLMs as part of the core methodology.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A More Information About the ArchCAD-400k

A.1 Detailed description of categorization

In Figure 8, we present some visual examples of content from each category. Engineering symbols from different categories can share great similarity. For example, simple rectangles can represent columns, holes, foundation, or furniture. Similarly, pairs of parallel lines might denote a pipeline, beam, or wall. At the same time, some instances can be drawn in diverse forms. As illustrated in Figure 8, there are at least six different ways to represent a door, and the appearance of stairs can vary greatly in different scenarios.

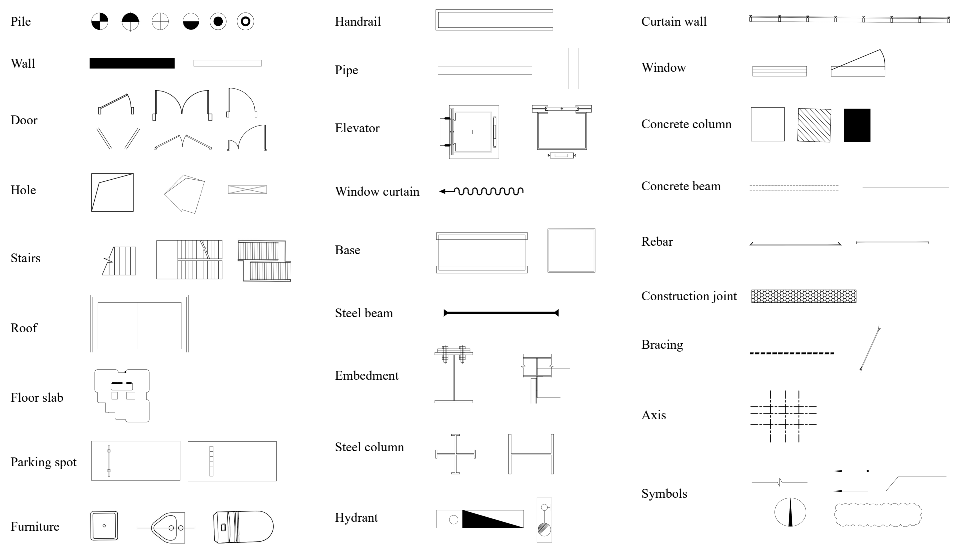


Figure 8: Visual examples of content in each category.

A.2 Additional information about the annotation pipeline

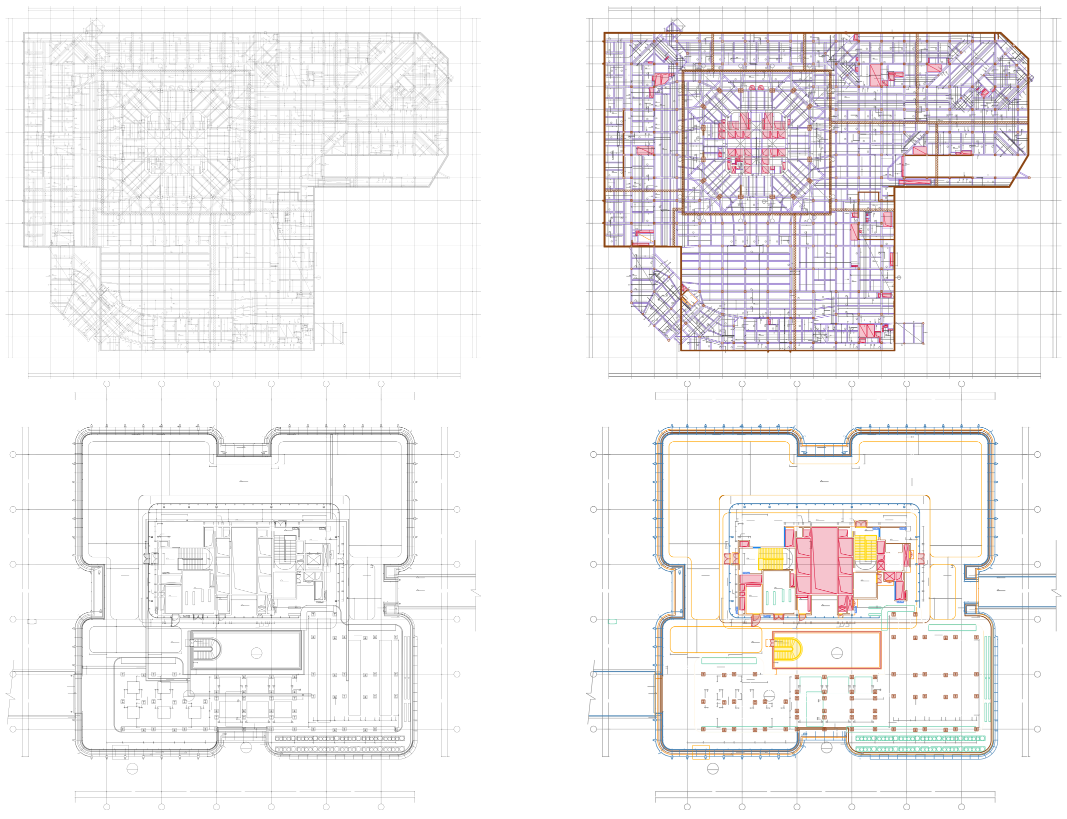
Our annotation pipeline adopts regex matching to map layer names to semantic classes. The layer names in standardized drawings have a hierarchical format like [Discipline]-[Category]-[Modifier]. In Table 6, we list part of the standardized naming table for doors, walls, and stairs. Categories can be matched through the keyword in the table.

Table 6: Examples of correspondence between layer names and semantics

Semantic label	Standard layer specifications	Content description
Door	A-DOOR	The surface structure of the door.
	A-DOOR-FRAM	The internal steel frame supporting the door.
	A-DOOR-HEAD	The line indicating the position of the door beam.
	A-DOOR-ROLL	The layout or design of the fireproof roller shutter.
	…	…
Wall	A-WALL-BLOK	Masonry block wall for structural stability.
	A-WALL-CONC	Concrete wall with high strength and fire resistance.
	A-WALL-STUD	Lightweight partition with stud framing and drywall.
	A-WALL-PRHT	Partial-height wall for spatial division.
	A-WALL-SCRN	Metal wall for electromagnetic shielding.
	A-WALL-FINI	Final surface treatment or cladding of a wall.
	A-WALL-INSU	Insulation layer for thermal or acoustic performance.
	A-WALL-TPTN	Partial-height wall for restroom privacy.
	A-WALL-EXPL	Reinforced wall to withstand blast pressures.
	S-WALL-LINE	The outline or framework of a wall in structural drawings.
	S-WALL-HATCH	Represents the filling or material pattern of a wall in structural drawings.
	…	…
Stairs	A-STRS-TREA	Stepping surface and vertical connector in stair construction.
	A-STRS-ESCL	Mechanical moving stairs for vertical transportation between floors.
	A-STRS-HRAL	Safety rails along stair edges for fall prevention.
	S-STRS-LINE	Outline representing stair geometry in structural drawings.
…	…	…

A.3 Additional examples of ArchCAD-400k

An example of well-labeled large architectural drawings exists in the main text. We further show two labeled drawings in our dataset in Figure 9. Each of them covers an area over 8000 m^2, with various types of components.



(a) Raw CAD Drawings

(b) Panoptic labeled Drawings

Figure 9: Additional example of the annotated drawings.

B More Evaluations

B.1 Analysis on Computational Efficiency

To evaluate the computational efficiency of the proposed method, we conducted additional experiments measuring inference latency across representative approaches. Our model operates on images with a resolution of 700×700, which introduces only a moderate computational overhead. Compared with SymPoint-V2 [13], the proposed method increases inference latency by approximately 30%, while achieving 3.0% and 10.1% higher panoptic quality on ArchCAD-400K and FloorPlanCAD, respectively. These results indicate that the proposed approach attains a favorable balance between computational efficiency and segmentation accuracy.

Table 7: Comparison with existing methods on FloorPlanCAD and ArchCAD-400K.

Method	PQ (FloorPlanCAD)	PQ (ArchCAD-400K)	Average Latency
CADTransformer [12]	68.9	60.0	159ms
SymPoint [14]	83.3	47.6	66ms
SymPoint-V2 [13]	83.2	60.5	95ms
DPSS (ours)	**86.2**	**70.6**	121ms

B.2 Additional Quantitative Evaluations

We present the detailed experimental results of various methods on the ArchCAD-400k, including the panoptic quality (PQ) for each category as well as the mean IoU (Intersection over Union) for each category. The IoU for each category is obtained by calculating the intersection over union between the predicted panoptic segmentation masks and the ground truth masks. Different methods excel at handling different types of objects, DPSS achieves higher PQ and IoU metrics. For the current spotting results, there is still significant room for improvement.

Table 8: Quantitative results for panoptic symbol spotting of each category on ArchCAD-400k

class	DPSS		CADTransformer [11]		SymPoint [14]		SymPointV2 [13]	
	IoU	PQ	IoU	PQ	IoU	PQ	IoU	PQ
symbol	84.4	89.8	62.4	70.7	76.4	80.9	74.2	79.3
axis	90.8	85.4	23.2	32.9	58.3	56.2	71.8	83.7
door	64.4	75.9	53.2	64.9	63.9	75.6	82.6	79.1
floor slab	40.8	85.6	14.4	80.3	20.9	82.8	47.7	69.4
elevator	45.6	77.6	26.5	63.9	54.6	73.3	28.0	69.2
stairs	60.4	65.4	35.3	45.2	41.3	55.4	43.7	78.0
furniture	72.5	88.7	66.5	73.3	63.2	83.9	43.4	54.0
hole	61.1	70.2	32.1	40.0	59.6	53.8	50.2	60.7
window	47.8	57.2	52.4	55.0	64.7	68.4	57.8	70.8
curtain wall	65.9	83.7	48.8	61.4	50.3	72.6	33.9	39.1
wall	73.4	75.6	39.1	45.9	52.7	57.7	42.3	70.7
concrete column	65.8	83.6	59.5	66.8	64.4	71.6	63.9	69.8
steel column	48.6	80.3	50.7	71.8	52.9	82.8	51.1	79.2
concrete beam	81.8	81.7	43.8	45.0	61.5	62.1	48.3	77.9
steel beam	79.1	73.0	19.7	34.9	47.5	52.2	71.5	77.4
parking spot	66.8	73.2	57.9	76.9	46.0	79.3	72.1	69.0
roof	3.5	28.7	3.7	39.1	5.7	50.1	55.6	76.1
base	85.0	90.3	65.8	75.2	71.2	83.1	16.3	21.3
bracing	19.6	33.8	7.8	20.0	20.2	23.2	67.6	81.1
rebar	72.7	93.1	39.4	69.4	56.5	81.1	53.5	75.9
equipment	37.2	45.8	1.0	7.0	30.8	31.8	28.4	29.1
handrail	73.6	65.9	31.2	31.0	40.2	42.7	70.3	85.8
pipe	44.4	48.3	32.9	34.4	51.7	41.7	52.5	42.4
window curtain	50.4	80.3	58.6	72.1	49.2	67.7	46.3	47.5
construction joint	0.0	5.3	0.0	0.3	2.5	4.8	4.1	3.7
embedment	0.0	0.0	0.0	0.0	0.0	0.0	1.2	2.8
hydrant	70.7	71.7	68.6	64.5	74.1	69.4	49.5	34.0
overall	70.6	67.04	47.6	49.1	60.5	59.1	60.0	60.8

C Limitations and Future Work

Although the proposed DPSS demonstrates strong performance on the panoptic symbol spotting task, some limitations remain. Notably, the current approach cannot process an entire vector drawing in a single pass, which leads to high computational overhead. With the availability of the ArchCAD-400k dataset, more complex and comprehensive research questions related to panoptic symbol spotting and the analysis of engineering line drawings can be explored. For example, future directions include the pre-trained models tailored for CAD vector graphics, and efficient inference strategies to handle large scale real-world drawings.