HIERARCHICAL PROTOTYPE LEARNING FOR 3D SCENE GRAPH GENERATION FROM RGB-D SEQUENCES

Anonymous authors
Paper under double-blind review

ABSTRACT

3D Scene Graph Generation (3DSGG) aims to construct structured scene representations by recognizing object nodes and predicting relational edges from 3D environments. Existing RGB-D sequence-based methods have made promising progress by reconstructing 3D geometry and enhancing node and edge features with multi-modal inputs. However, they still struggle to address the severe inter-class similarity and intra-class diversity of both objects and relations in complex 3D scenes. In this work, we propose HPSSG, a hierarchical prototype learning framework for RGB-D sequence-based 3DSGG, inspired by the hierarchical perception process of human scene understanding. Specifically, semantic-level prototypes serve as stable semantic anchors to improve the separability between similar classes, while mode-level prototypes capture multiple latent patterns within each category to model diverse object instances and relation configurations. To maintain hierarchical consistency, we further introduce hierachy-aware prototype entailment learning, which constrains mode-level prototypes with their corresponding category semantics. Moreover, prior-guided compositional prototype learning uses subject-object compositional priors to select and refine mode-level edge prototypes, improving the structural consistency of relation prediction. Experiments on the 3DSSG dataset demonstrate that our method outperforms various leading methods without any additional inference cost. Our code will be released.

1 INTRODUCTION

In recent years, scene graphs (SG) have received increasing attention in visual understanding as a structured representation of scenes. Instead of representing a scene as a holistic feature, SG is typically defined as a graph-structured representation, where nodes denote object instances in the scene and edges describe semantic relations between objects or between objects and their surrounding environment. Early SG studies mainly focus on 2D images [1, 2], and are later extended to video understanding [3, 4] to capture more complex temporal relations and dynamic interactions. With the development of 3D perception, SG is further introduced into 3D tasks [5, 6], where they serve as an important intermediate representation for robot navigation [7, 8], robot localization [9, 10], scene reasoning [11, 12] and 3D scene reconstruction [13, 14].

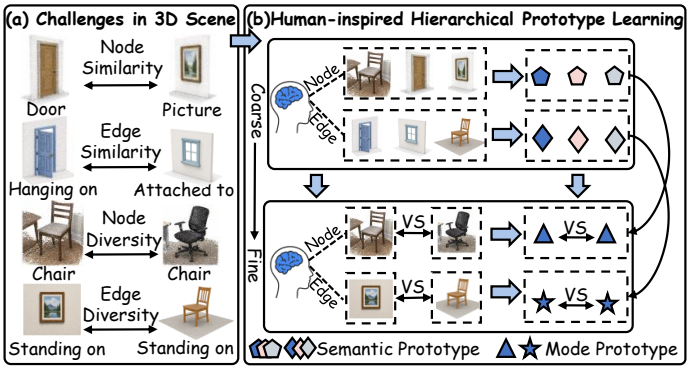

Figure 1: (a) The illustration of nodes and edges with intra-class diversity and inter-class similarity. (b) The illustration of Human Hierarchical Perception-inspired Dual-level Prototype Learning.

Many existing 3D scene graph generation (3DSGG) methods build scene representations from LiDAR point clouds [15, 16]. Although LiDAR data provides accurate geometry, its reliance on dedicated sensors and limited texture information restrict its practical use in real-world scenarios. As a result,

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

3DSGG from RGB or RGB-D image sequences is receiving increasing attention [17, 18]. Under this setting, existing methods usually reconstruct 3D scene from image observations and incorporate multi-modal cues (*e.g.*, visual features [5, 18] and textual information [19]) to enhance node and edge representations. To further improve relational reasoning, recent studies also introduce external priors as additional guidance for prediction (*e.g.*, statistical co-occurrence patterns [18] and historical knowledge [19]). Despite these notable advances, existing methods still struggle to fully address the two inherent limitations, as shown in Fig. 1a. First, **inter-class similarity.** Different node classes may share similar geometric shapes or appearance structures. For example, ***door*** and ***picture*** both exhibit planar rectangular forms, which can easily lead to confusion in node classification. A similar issue also appears in edge, where different edge classes may correspond to close spatial layouts or contact patterns. For example, ***hanging on*** and ***attached to*** both describe a tight dependency between an object and a supporting surface, making edge classification more ambiguous. Second, **intra-class diversity.** Node from the same class often show large variations in shape and scale in real 3D scenes. For example, different ***chair*** may have different geometries. Similarly, the same edge class may occur under diverse object pairs and spatial configurations, where ***standing on*** can describe different support patterns between supporting and supported objects in real-world indoor environments.

To address the above challenges caused by inter-class similarity and intra-class diversity, we draw inspiration from the **hierarchical understanding process** of human scene perception. When understanding complex scenes, humans typically first form coarse-grained category awareness and then progressively leverage geometric and visual cues for fine-grained structural discrimination, as shown in Fig. 1b. This coarse-to-fine perception process suggests that effective 3DSGG should not rely solely on a single category-level representation, but instead requires a hierarchical representation that can simultaneously preserve stable category semantics and capture diverse intra-class structures. Based on this observation, we propose a **semantic-to-mode hierarchical prototype learning** paradigm, where semantic-level prototypes serve as stable category anchors, while mode-level prototypes capture diverse latent structural patterns within each category under the guidance of semantic-level prototypes. This paradigm improves the separability of nodes and edges while preserving their intra-class structural diversity, enabling the model to adapt to complex 3D scenes more effectively in practice.

In practice, after feature extraction, we shape the feature space with semantic-level and mode-level prototypes rather than a single center. To this end, we design several complementary learning mechanisms. ❶ **Hierarchical prototype learning,** where **semantic-level prototypes** provide stable class anchors to establish reliable category-level semantic boundaries and improve global separability (§3.3), while **mode-level prototypes** capture diverse latent intra-class patterns under each category to model fine-grained structural variations and enhance local discriminability (§3.4). ❷ **Hierarchy-aware prototype entailment learning** to maintain the semantic consistency between semantic-level and mode-level prototypes, preventing mode-level prototypes from drifting away from their corresponding class semantics (§3.5). ❸ **Prior-guided compositional prototype learning,** which leverages subject-object mode-level prototype representations as informative priors to guide the selection of mode-level edge prototypes, thereby explicitly modeling node-edge compatibility and improving structural consistency in complex 3D scenes (§3.6).

In summary, our contributions are summarized as follows: **First,** we propose a hierarchical prototype representation learning framework for 3DSGG. This framework uses semantic-level prototypes to model stable category semantics and mode-level prototypes to characterize diverse intra-class latent patterns, thereby jointly alleviating inter-class similarity and intra-class diversity. **Second,** We design hierarchy-aware prototype entailment learning to keep mode-level prototypes semantically aligned with their corresponding semantic-level prototypes while preserving intra-class variations. **Third,** We introduce prior-guided compositional prototype learning that leverages subject-object mode-level prototype to guide edge mode-level prototypes, thereby enabling more robust edge prediction. **Finally,** our proposed method serves only as an auxiliary module during training. During inference, these modules can be removed, thus introducing no additional parameters or inference latency.

To comprehensively evaluate our method HPSSG, we conduct experiments on the widely used 3DSSG benchmark. Experimental results show that, using point clouds reconstructed from RGB-D sequences by our pipeline, HPSSG consistently outperforms existing state-of-the-art methods in both node and edge prediction. Specifically, our method achieves superior performance on key metrics, with Tri. reaching **44.8**% (+**4.3%**) and Obj. to **65.1**(+**3.3**%). We also conduct experiments using ground-truth point clouds, where Tri. reaches **74.2**% (+**4.5**%). In addition, a series of ablation studies further verifies the effectiveness of each core component and the overall design of HPSSG.

2 Related Work

3D Scene Graph Generation (3DSGG). According to the input modality, existing 3DSGG methods can be broadly divided into point cloud-based methods and image sequence-based methods. Early point cloud-based 3DSGG methods mainly follow two research directions. The first direction focuses on contextual dependency modeling for relation prediction [20, 21, 22, 16, 23, 24, 25, 26]. These methods usually incorporate multi-dimensional edge features and propagate information between nodes and edges to explicitly capture contextual dependencies. The second direction introduces structural priors and spatial reasoning [27, 28], leveraging the inherent hierarchy, layout constraints, and regularities of indoor environments to regularize relation prediction. Recently, driven by the rapid progress of vision-language models, 3DSGG has also been extended to open-vocabulary settings [29, 30, 31, 32], where cross-modal alignment is used to inject textual semantics into 3D geometric representations. However, in real-world scenarios, acquiring high-quality point clouds is often limited by occlusions. This has motivated increasing attention to image sequence-based 3DSGG.

Existing image sequence-based methods typically reconstruct 3D geometry from RGB/RGB-D sequences before scene graph generation and can be categorized into two paradigms. Representation enhancement methods [5, 6, 33, 17] exploit multi-modal cues to learn discriminative node and edge features, while inference refinement methods [18, 19] leverage priors or external knowledge to refine predictions. Recently, 2D-to-3D lifting strategies [34] have been explored. However, existing methods still overlook the intra-class diversity and inter-class similarity of nodes and edges.

Prototype Learning. Prototype learning is inspired by the observation that humans often rely on prototypical knowledge for recognition and reasoning [35, 36]. Unlike conventional Softmax-based classifiers [37, 38], prototype-based methods [39, 40, 36, 41] make predictions by measuring the distance or similarity between sample features and class prototypes. A typical formulation represents each category by the centroid of its samples [42], which provides a simple and interpretable way to organize the feature space. Due to its exemplar-driven nature, prototype learning has been widely explored in supervised representation learning [43, 44], few-shot learning [45, 42], and zero-shot learning [36, 46]. These studies show that prototypes can act not only as classification references, but also as structural constraints that enhance intra-class compactness and inter-class separability.

3 Methodology

3.1 Problem Definition

Formally, given an RGB-D image sequence, the objective of 3D scene graph generation (3DSGG) can be defined as:

$$\mathcal{S} = (\mathcal{V}, \mathcal{E}), \tag{1}$$

where $\mathcal{V} = \{y_i\}_{i=1}^{M}$ and $\mathcal{E} = \{z_{i \to j}\}$ denote the object nodes and directed relation edges over ordered object pairs (i, j), respectively. Each node and edge is associated with an object category label $y_i \in \{1, \ldots, c\}$ and a relation label $z_{i \to j} \in \{1, \ldots, r\}$.

3.2 Overview

The overall framework is illustrated in Fig. 2. Given an RGB-D sequence, we follow SceneGraphFusion [33] to reconstruct dense 3D geometry and obtain instance-level point clouds. Multi-view RGB observations are also extracted for each object. The point clouds and images are then fed into PointNet [47] and DINOv2 [48] to extract and enhance node and edge features for final prediction.

To address inter-class similarity and intra-class diversity, we design a **hierarchical prototype learning** composed of semantic-level and mode-level prototypes. **Semantic-level prototypes** serve as stable global semantic anchors and improve category separability through prototype-anchored contrastive learning (§3.3). **Mode-level prototypes** are derived from semantic-level prototypes to capture diverse intra-class patterns, and further enhance the local discriminative structure of the feature space through pattern-aware prototype assignment and scatter-guided discriminative learning (§3.4). In addition, **prototype entailment learning** maintains semantic consistency between the two prototype levels (§3.5), while **prior-guided compositional prototype learning** uses node-pair prototype representations to constrain edge prototypes, improving the structural consistency of relation representations (§3.6). Finally, the optimized node and edge features are used for prediction.

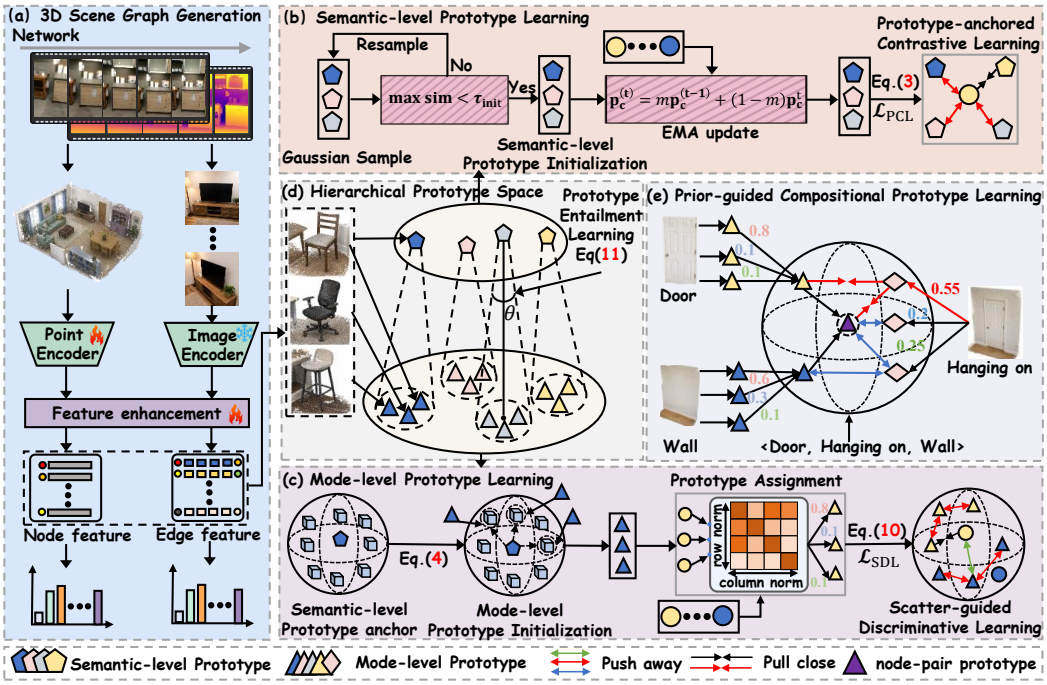

Figure 2: The overall of HPSSG. (a) We guide the optimization of node and edge feature spaces through (d) hierarchical prototype learning, which consists of (b) Semantic-level Prototype Learning (§3.3) and (c) Mode-level Prototype Learning (§3.4), aiming to address inter-class similarity and intra-class diversity. Meanwhile, (d) Prototype Entailment Learning (§3.5) and (e) Compositional Prototype Learning (§3.6) further constrain the prototype space by hierarchical and prior knowledge.

3.3 Semantic-level Prototype Learning

To provide stable semantic anchors for nodes and edges, we introduce semantic-level prototypes, as shown in Fig. 2(b). Each category is assigned one prototype to represent its global semantic center, which helps reduce inter-class ambiguity caused by similar geometry and spatial layouts while providing category-level structural guidance. For clarity, we use the node branch as an example in the following, while the edge branch follows the same process.

Semantic-level Prototype Initialization. To initialize these semantic-level prototype sets, we independently construct prototype banks for nodes and edges. For each category, a candidate prototype is sampled from a Gaussian distribution and then normalized onto the unit hypersphere. To avoid overly similar initial prototypes across different categories, we compute the pairwise cosine similarities among prototypes and require the maximum inter-category similarity to be lower than a predefined threshold τ_{init}. If this constraint is not satisfied, the prototypes are resampled. This initialization strategy encourages an initially separated prototype space and provides stable semantic anchors for subsequent feature optimization.

Semantic-level Prototype Update. Since 3D scene graph data usually exhibit a pronounced long-tailed distribution, category frequencies can be highly imbalanced within each mini-batch. Directly replacing semantic-level prototypes with mini-batch statistics may make them sensitive to sampling bias and noisy samples. Therefore, we update the prototypes using an exponential moving average. For a node category appearing in the current mini-batch, we first compute its batch-wise prototype $\bar{\boldsymbol{p}}_c^n$ by averaging the features of all node samples from category. The corresponding node semantic-level prototype $\boldsymbol{p}_c^n$ is then updated as:

$$\boldsymbol{p}_c^n \leftarrow \mu \boldsymbol{p}_c^n + (1-\mu)\bar{\boldsymbol{p}}_c^n, \tag{2}$$

where $\mu \in [0, 1)$ denotes the momentum coefficient. Prototypes of categories absent from the current mini-batch remain unchanged. Edge semantic-level prototypes are updated in the same manner.

Prototype-anchored Contrastive Learning. After obtaining the semantic-level prototypes, we introduce a prototype-anchored contrastive learning to explicitly shape the category structure of node and edge feature spaces. For each feature, its corresponding semantic-level prototype is treated as the positive anchor $\boldsymbol{p}_+^n$, while prototypes from other categories are regarded as negative anchors $\mathcal{P}_-$. In

this way, the feature is encouraged to be close to the semantic center of its ground-truth category and distant from irrelevant category prototypes. For a node feature $\boldsymbol{x}_i^n$, which can be defined as:

$$\mathcal{L}_{\mathrm{PCL}} = -\frac{1}{|\mathcal{B}^n|}\sum_{i\in\mathcal{B}^n}\log\frac{\exp\left(\mathrm{sim}(\boldsymbol{x}_i^n,\boldsymbol{p}_+^n)/\tau\right)}{\exp\left(\mathrm{sim}(\boldsymbol{x}_i^n,\boldsymbol{p}_+^n)/\tau\right)+\sum_{\boldsymbol{p}_-^n\in\mathcal{P}_-}\exp\left(\mathrm{sim}(\boldsymbol{x}_i^n,\boldsymbol{p}_-^n)/\tau\right)}. \tag{3}$$

where $\mathcal{B}^n$ denotes the number of features in the mini-batch, $\tau > 0$ is the temperature coefficient and $\mathrm{sim}(\cdot,\cdot)$ denotes cosine similarity. Unlike sample-wise contrastive learning, our method contrasts features with semantic-level prototypes rather than mini-batch samples. By avoiding unreliable pair construction under long-tailed category distributions, it provides more stable semantic-level supervision for both nodes and edges and improves the separability of ambiguous categories.

3.4 Mode-level Prototype Learning

Although semantic-level prototypes provide stable global semantic anchors, a single class center is insufficient to capture complex intra-class feature distributions in 3D scenes. The same object class may exhibit different geometric shapes and scales. If all samples from the same class are constrained by only one semantic-level prototype, their features may be over-compressed around a single semantic center, which weakens the representation of intra-class structures. We therefore introduce mode-level prototypes to model diverse local semantic modes within each category, as shown in Fig. 2(d).

Mode-level Prototype Initialization. To obtain stable and diverse mode-level prototypes at the beginning of training, we use semantic-level prototypes as anchors and initialize mode-level prototypes around them. For each class c, given the semantic-level prototype $\boldsymbol{p}_c$, we sample a tangent-space perturbation $\tilde{\boldsymbol{\epsilon}}_{c,k}$ orthogonal to $\boldsymbol{p}_c$ for the k-th mode-level prototype $\boldsymbol{q}_{c,k}$, and initialize it as:

$$\boldsymbol{q}_{c,k} = \mathrm{Norm}\left(\boldsymbol{p}_c + \sigma\,\mathrm{Norm}(\tilde{\boldsymbol{\epsilon}}_{c,k})\right), \tag{4}$$

where σ controls the initial dispersion of mode-level prototypes around the semantic-level prototype. In this way, mode-level prototypes within the same class share consistent category-level semantics while preserving local diversity, providing a stable initialization for subsequent prototype learning.

Pattern-aware Prototype Assignment. To discover diverse intra-class patterns, we perform mode-level prototype assignment within each ground-truth class during training. For node class c, we augment its mini-batch features $\widetilde{\boldsymbol{X}}_c^n$ with a category-conditioned memory $\boldsymbol{M}_c^n$ to alleviate batch instability under long-tailed distributions, forming $\boldsymbol{Z}_c^n = [\widetilde{\boldsymbol{X}}_c^n; \boldsymbol{M}_c^n]$. Given the mode-level prototypes $\boldsymbol{Q}_c^n \in \mathbb{R}^{K\times d_n}$, we learn a soft assignment matrix $\boldsymbol{A}_c^n \in \mathbb{R}_+^{\bar{N}_c\times K}$ to assign features in $\boldsymbol{Z}_c^n$ to K prototypes, formulated as an optimal transport problem:

$$\begin{aligned}&\max_{\boldsymbol{A}_c^n}\left\langle \boldsymbol{A}_c^n, \boldsymbol{Z}_c^n(\boldsymbol{Q}_c^n)^\top\right\rangle + \varepsilon\boldsymbol{H}(\boldsymbol{A}_c^n),\\ &s.t.\quad \boldsymbol{A}_c^n\mathbf{1}_K = \mathbf{1}_{\bar{N}_c},\quad (\boldsymbol{A}_c^n)^\top\mathbf{1}_{\bar{N}_c} = \frac{\bar{N}_c}{K}\mathbf{1}_K,\quad \boldsymbol{A}_c^n \geq 0.\end{aligned} \tag{5}$$

Here, $\langle\cdot,\cdot\rangle$ denotes the Frobenius inner product, $\boldsymbol{H}(\boldsymbol{A}_c^n)$ is the entropy regularizer, and ε controls the assignment smoothness. The row constraint normalizes the assignment of each feature, while the column constraint encourages balanced activation of different mode-level prototypes. In practice, the above problem is efficiently solved by Sinkhorn-Knopp iterations [49, 50]:

$$\boldsymbol{A}_c^n = \mathrm{diag}(\boldsymbol{u})\exp\left(\frac{\boldsymbol{Z}_c^n(\boldsymbol{Q}_c^n)^\top}{\varepsilon}\right)\mathrm{diag}(\boldsymbol{v}), \tag{6}$$

where $\boldsymbol{u} \in \mathbb{R}^{\bar{N}_c}$ and $\boldsymbol{v} \in \mathbb{R}^K$ are normalization vectors iteratively updated to satisfy the row and column constraints. Since the assignment is restricted to the ground-truth category, features in $\boldsymbol{Z}_c^n$ are only matched with prototypes in $\boldsymbol{Q}_c^n$. This design allows mode-level prototypes to capture category-specific intra-class diversity under the guidance of semantic-level prototypes.

Scatter-guided Discriminative Learning. Although pattern-aware assignment links features with mode-level prototypes, it lacks explicit control over their global distribution. Since pairwise distance constraints struggle to capture distributional dispersion and prevent intra-class prototype collapse [51, 52], we introduce scatter matrices to impose distribution-level discriminative constraints on feature and prototype spaces through three scatter analyses as follow:

Feature-Prototype Intra-class Scatter. To encourage each feature to approach its assigned mode-level prototype, we measure the deviation between each node feature and its assignment-weighted prototype. The feature-prototype intra-class scatter is defined as:

$$\mathbf{\Omega}_w^n = \frac{1}{|\mathcal{B}^n|} \sum_{i \in \mathcal{B}^n} (\widetilde{\boldsymbol{x}}_i^n - \bar{\boldsymbol{q}}_i^n)(\widetilde{\boldsymbol{x}}_i^n - \bar{\boldsymbol{q}}_i^n)^\top, \tag{7}$$

where $\mathcal{B}^n$ denotes the number of node features in the mini-batch, $\widetilde{\boldsymbol{x}}_i^n$ is the i-th node feature, $\bar{\boldsymbol{q}}_i^n$ denote the assignment-weighted prototype.

Feature-Prototype Inter-class Scatter. Although intra-class scatter pulls features closer to their assigned prototypes, it still lacks cross-category separation constraints. To address this, for each node feature, we select the most similar mode-level prototype from other categories as its nearest negative prototype, and define the feature-prototype inter-class scatter accordingly:

$$\mathbf{\Omega}_b^n = \frac{1}{|\mathcal{B}^n|} \sum_{i \in \mathcal{B}^n} \left(\widetilde{\boldsymbol{x}}_i^n - \boldsymbol{q}_i^{n,-}\right)\left(\widetilde{\boldsymbol{x}}_i^n - \boldsymbol{q}_i^{n,-}\right)^\top, \tag{8}$$

where $\boldsymbol{q}_i^{n,-}$ is its nearest negative mode-level prototype from other categories.

Prototype-Prototype Intra-class Scatter. Feature-prototype scatters regulate their relations but cannot explicitly prevent intra-class mode-level prototype collapse. To preserve mode diversity, we take the most similar prototype within the same category as the nearest neighbor $\boldsymbol{q}_{c,k}^{n,+}$ for each prototype, and define the prototype-prototype intra-class scatter as:

$$\mathbf{\Omega}_p^n = \frac{1}{CK} \sum_{c=1}^{C} \sum_{k=1}^{K} \left(\boldsymbol{q}_{c,k}^n - \boldsymbol{q}_{c,k}^{n,+}\right)\left(\boldsymbol{q}_{c,k}^n - \boldsymbol{q}_{c,k}^{n,+}\right)^\top, \tag{9}$$

where C and K denote the number of categories and the number of mode-level prototypes per category, respectively. $\boldsymbol{q}_{c,k}^n$ is the k-th mode-level node prototype of category c.

Optimization Objective. Based on the above scatter matrices, our final objective is to minimize Eq. 7 while maximizing Eq. 8 and Eq. 9. Inspired by Linear Discriminant Analysis (LDA) [53, 54], the scatter-guided discriminative learning is defined in a ratio-trace form:

$$\mathcal{L}_{\text{SDL}} = \left[\text{Tr}\left((\mathbf{\Omega}_w^n + \gamma \boldsymbol{I})^{-1} \mathbf{\Omega}_b^n\right) + \text{Tr}\left((\mathbf{\Omega}_w^n + \gamma \boldsymbol{I})^{-1} \mathbf{\Omega}_p^n\right)\right]^{-1}, \tag{10}$$

where γ is used for numerical stability and $\boldsymbol{I}$ denotes the identity matrix for numerical stability.

3.5 Hierarchy-aware Prototype Entailment Learning

Semantic-level and mode-level prototypes encode different levels of semantics. For category c, the semantic-level prototype $\boldsymbol{p}_c^n$ captures the global category direction, while mode-level prototypes $\boldsymbol{q}_{c,k}^n$ model diverse intra-class patterns. To preserve hierarchical consistency and avoid semantic drift, we introduce hierarchy-aware prototype entailment learning, which constrains each mode-level prototype within the semantic cone defined by its semantic-level prototype. After normalization, a mode-level prototype is considered semantically consistent if $(\hat{\boldsymbol{q}}_{c,k}^n)^\top \hat{\boldsymbol{p}}_c^n \geq \cos\tau_n$, where τ_n denotes the cone half-angle. We therefore define the prototype entailment learning as follows:

$$\mathcal{L}_{\text{PEL}} = \frac{1}{CK} \sum_{c=1}^{C} \sum_{k=1}^{K} \max\left(0, \cos\tau_n - (\hat{\boldsymbol{q}}_{c,k}^n)^\top \hat{\boldsymbol{p}}_c^n\right), \tag{11}$$

where C and K denote the number of categories and the number of mode-level prototypes per category, respectively. In this way, the hierarchy-aware prototype entailment learning enhances semantic-to-mode hierarchical consistency while maintaining intra-class diversity.

3.6 Prior-guided Compositional Prototype Learning

Although node and edge mode-level prototypes capture intra-class diversity for object and relation categories, they are learned independently. However, edge semantics in 3D scene graphs are highly conditioned on the compositional semantics of the two endpoint nodes. We therefore introduce prior-guided compositional prototype learning, as shown in Fig. 2(e), which uses subject-object prototype conditions to guide edge mode-level prototype selection.

For a relation triplet $\langle v_i, z_{i\to j}, v_j \rangle$, we obtain the matched node mode-level prototypes $\boldsymbol{q}_i^n$ and $\boldsymbol{q}_j^n$ for the subject and object nodes. We then project them into three normalized conditional embeddings,

Table 1: **Quantitative results** (§4.2) are evaluated on 3DSSG [55] using dense point clouds. Best and second-best results are bold and underlined. ∗: without *none* relationship. †: Using ResNet-18 as image encoder.

Method	Input	Recall(%)						mRecall(%)	
		Tri.↑	Tri.*↑	Obj.↑	Obj.*↑	Pred.↑	Pred.*↑	Obj.↑	Pred.↑
IMP [56] [CVPR2017]	Dense	25.4	14.4	47.7	48.1	89.9	16.8	35.8	19.9
VGfM [57] [ACCV2018]	Dense	27.4	17.0	50.1	51.3	**90.6**	19.8	33.6	22.3
SGPN [55] [CVPR2020]	Dense	13.0	11.9	35.6	36.2	86.8	25.7	26.0	24.3
SGFN [33] [CVPR2021]	Dense	29.9	21.5	52.2	53.8	89.5	24.6	35.9	27.1
MonoSSG [5]† [CVPR2023]	Dense	32.9	23.1	54.5	55.1	88.7	26.6	45.4	35.2
MonoSSG [5] [CVPR2023]	Dense	35.0	26.9	59.6	61.0	86.9	26.9	55.9	38.6
OCRL [16] [NeurIPS2025]	Dense	30.0	23.7	52.5	53.9	88.6	26.4	40.4	27.9
HEDSGP [17] [TPAMI2025]	Dense	31.3	-	55.5	-	90.1	-	47.3	36.9
SCRSSG [18] [ICCV2025]	Dense	<u>40.5</u>	25.7	<u>61.8</u>	<u>61.1</u>	90.4	27.6	**60.5**	<u>39.2</u>
HPSSG (Ours) †	Dense	35.8	<u>27.0</u>	58.2	59.9	89.8	<u>29.3</u>	48.5	37.1
HPSSG (Ours)	Dense	**44.8**	**29.3**	**65.1**	**66.8**	<u>90.5</u>	**30.5**	<u>60.0</u>	**41.8**

including the subject condition $\boldsymbol{u}_i^s = W_s \boldsymbol{q}_i^n$, the object condition $\boldsymbol{u}_j^o = W_o \boldsymbol{q}_j^n$, and the directional condition $\boldsymbol{u}_{i,j}^d = W_d(\boldsymbol{q}_j^n - \boldsymbol{q}_i^n)$, where W_s, W_o, and W_d are projections. For the ground-truth edge category r, let $\boldsymbol{q}_{r,k}^e$ denote the k-th mode-level edge prototype. The compatibility score $\psi_{i,j,r,k}$ is computed as the sum of cosine similarities between $\boldsymbol{q}_{r,k}^e$ and the three conditional embeddings. We select the edge prototype with the highest compatibility as the positive anchor, denoted as $\boldsymbol{q}_{r,k^+}^e$, and treat the remaining edge prototypes within the ground-truth edge category r as negative anchors. The prior-guided compositional prototype learning is defined as:

$$\mathcal{L}_{\text{CPL}} = -\frac{1}{|\mathcal{T}|} \sum_{(i,r,j)\in\mathcal{T}} \log \frac{\exp(\psi_{i,j,r,k^+}/\tau_e)}{\exp(\psi_{i,j,r,k^+}/\tau_e) + \sum_{k\neq k^+} \exp(\psi_{i,j,r,k}/\tau_e)}, \tag{12}$$

where τ_e is a temperature parameter. This objective uses subject-object compositional conditions to guide the selection and refinement of mode-level edge prototypes within the ground-truth category.

4 Experiments

4.1 Experimental Setup

Dataset. We conduct experiments on the widely used 3DSGG benchmark 3DSSG [55]. Built upon the 3RScan dataset [58], 3DSSG provides annotated 3D semantic scene graphs. We report results using both reconstructed dense point clouds (Dense) and ground-truth point clouds (GT) as inputs. For the Dense setting, following the ScanNet benchmark, we map the labels to a subset of 20 node classes and 8 edge classes. For the GT setting, we evaluate our model with this same mapping.

Evaluation Metric. We evaluate our model across three standard task dimensions: (1) Triplet Classification, (2) Object Classification, (3) Predicate Classification. Following the standard evaluation protocol [5], we adopt Top-1 Recall (R@1) and Mean Recall@1 (mR@1) as our primary metrics.

Implementation Details. All experiments are conducted on a single NVIDIA GeForce RTX 4090 GPU with a batch size of 1, using an early-stopping criterion implemented in PyTorch. HPSSG is trained end-to-end with AdamW optimizer, starting from a learning rate of 1×10^{-4}.

Overall Loss Function. The training total loss $\mathcal{L}$ of five components, including the losses defined in Eq. 3, Eq. 10, Eq. 11, and Eq. 12, and the cross-entropy loss $\mathcal{L}_{\text{CE}}$ for node and edge classification. Specifically, the overall loss is computed as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{CE}} + \lambda_2 \mathcal{L}_{\text{PCL}} + \lambda_3 \mathcal{L}_{\text{SDL}} + \lambda_4 \mathcal{L}_{\text{PEL}} + \lambda_5 \mathcal{L}_{\text{CPL}}. \tag{13}$$

The above loss $\mathcal{L}$ is applied to both nodes and edges, where λ_1 is dynamically computed based on the number of nodes and edges in the current scene, $\lambda_2 = 0.05$, $\lambda_3 = 0.3$, $\lambda_4 = 0.02$, and $\lambda_5 = 0.03$.

4.2 Comparison to State-of-the Arts

Performance on Dense Point Cloud Setting. As shown in Table 1, under the dense point cloud setting, our method outperforms state-of-the-art approaches on multiple metrics. Specifically, Tri. and Obj. reach **44.8** (**+4.3**%) and **65.1** (**+3.3**%). After excluding the none relationship, Tri.* and Obj.* further improve to **29.3** (**+2.4**%) and **66.8** (**+5.7**%). Moreover, even when using a lightweight ResNet-18 image encoder, our method remains competitive, indicating that the performance gains

Table 2: **Quantitative results** (§4.2) are evaluated on 3DSSG [55] with GT point clouds. Best and second-best results are bold and underlined. ⋆: without *none* relationship. †: Using ResNet-18 as image encoder.

Method	Input	Recall(%)						mRecall(%)	
		Tri.↑	Tri.*↑	Obj.↑	Obj.*↑	Pred.↑	Pred.*↑	Obj.↑	Pred.↑
IMP [56] [CVPR2017]	GT	45.3	44.3	65.4	65.9	94.0	56.6	56.2	41.8
VGfM [57] [ACCV2018]	GT	52.9	51.4	70.7	71.3	94.9	62.8	59.5	46.8
SGPN [55] [CVPR2020]	GT	31.8	39.7	55.1	55.6	95.4	71.0	47.7	61.5
SGFN [33] [CVPR2021]	GT	42.6	47.6	63.6	64.4	94.2	68.9	53.6	63.1
VL-SAT [22] [CVPR2023]	GT	66.1	-	78.4	-	89.9	-	53.8	67.1
MonoSSG [5]† [CVPR2023]	GT	63.8	63.4	79.3	80.0	95.6	76.0	78.2	64.8
MonoSSG [5] [CVPR2023]	GT	<u>69.7</u>	<u>65.5</u>	81.7	<u>82.5</u>	95.8	77.0	79.7	<u>72.9</u>
EgoSG [6] [CVPRW2024]	GT	61.1	-	56.4	-	78.6	-	48.6	66.7
OCRL [16] [NeurIPS2025]	GT	42.6	48.0	62.3	62.8	93.6	71.4	62.3	67.7
HESGR [19] [TCVST2025]	GT	67.6	-	<u>83.3</u>	-	**96.5**	-	69.1	71.2
HEDSGP [17] [TPAMI2025]	GT	66.1	-	83.2	-	<u>96.1</u>	-	78.9	68.8
HPSSG(Ours)†	GT	68.3	64.7	81.2	81.9	96.0	<u>77.5</u>	<u>80.0</u>	68.2
HPSSG (Ours)	GT	**74.2**	**69.7**	**84.9**	**85.8**	96.0	**78.1**	**82.9**	**74.7**

Table 3: **Analysis of core components** (§4.3) on 3DSSG [55] within 20 node and 8 edge classes, which from reconstructed dense point clouds.

Module	$\mathcal{L}_{\text{PCL}}$ (Eq. 3)	$\mathcal{L}_{\text{SDL}}$ (Eq. 10)	$\mathcal{L}_{\text{PEL}}$ (Eq. 11)	$\mathcal{L}_{\text{CPL}}$ (Eq. 12)	Recall(%)			mRecall(%)	
					Tri.↑	Obj.↑	Pred.↑	Obj.↑	Pred.↑
Baseline(*w/o* Prototype Learning)					32.9	54.5	88.7	45.4	35.2
+ Semantic-level Prototype Learning (SPL)	✓				33.3	56.2	87.8	45.9	35.5
+ Mode-level Prototype Learning (MPL)		✓			34.5	57.0	88.7	46.9	35.9
+ Semantic-Mode level Prototype Learning	✓	✓			33.9	56.5	88.8	45.9	35.2
+ Prototype Entailment Learning (PEL)	✓	✓	✓		35.3	57.6	89.3	47.3	36.8
+ Compositional Prototype Learning (CPL)	✓	✓	✓	✓	**35.8**	**58.2**	**89.8**	**48.5**	**37.1**

mainly stem from the proposed hierarchical prototype learning framework, which effectively models intra-class diversity and enhances the discrimination between semantically similar categories.

Performance on GT Point Cloud Setting. Table 2 report quantitative comparisons using ground-truth point clouds as input. In the Table 2, our method achieves the best performance on Tri. and Tri.*, reaching **74.2** (+**4.5**%) and **69.7** (+**4.2**%). Regarding the mRecall metric, consistently outperforming prior methods, our model also achieves the best results, reaching **82.9** (+**3.2**%) and **74.7** (+**1.8**%).

4.3 Ablation Study

We conduct ablation studies on 3DSSG [55] under the dense point cloud setting to evaluate each component. PointNet and ResNet-18 are used as point cloud and image encoders, respectively.

Key Component Analysis. We evaluate SPL (§3.3), MPL (§3.4), PEL (§3.5), and CPL (§3.6) in Table 3. SPL and MPL alone raise Tri. to **33.3**% and **34.5**%, respectively, validating semantic anchoring and intra-class mode modeling. Their direct combination reduces Tri. to 33.9%, suggesting the need for hierarchical coordination. Adding PEL improves all metrics by enforcing hierarchical consistency. Finally, CPL raises Tri. and Obj. to **35.8**% and **58.2**%, exceeding the baseline by **2.9**% and **3.7**%, demonstrating the benefits of subject-object prototype priors for edge prototype learning through explicit node-edge compatibility modeling in complex indoor 3D environments.

Number of Mode-level Prototypes. Table 4a studies the effect of prototype number. Both $K = 5$ and $K = 8$ outperform the baseline and the single-prototype setting, confirming the benefit of multiple intra-class prototypes. However, $K = 8$ slightly degrades performance compared with $K = 5$, suggesting that excessive prototypes may cause redundant partitions or unstable assignments.

Semantic-level Prototype Updating Coefficient. We analyze the effect of μ in Table 4b. Direct batch-wise updating with $\mu = 0$ performs worst. Increasing μ improves performance, and $\mu = 0.99$ achieves the best overall results, with Tri. and mRecall Obj. reaching **35.8**% and **48.5**%. A further increase to $\mu = 0.999$ slightly reduces performance due to limited adaptability.

Prototype Learning Branch. Table 4c shows that node and edge prototypes contribute differently. Node prototypes mainly improve object recognition, increasing Obj. from 54.5% to **57.2**%, while edge prototypes mainly enhance relation prediction, raising mRecall Pred. from 35.2% to **37.0**%. Using both branches achieves the best overall performance, demonstrating their complementarity.

Table 4: **A set of ablation studies** (§4.3) on 3DSSG [55], which using reconstructed dense point clouds.

Number of Prototypes	Recall(%) Tri.↑	Obj.↑	Pred.↑	mRecall(%) Obj.↑	Pred.↑
$K = 0$	32.9	54.5	88.7	45.4	35.2
$K = 1$	33.3	56.2	87.8	45.9	35.5
$K = 5$	**35.8**	**58.2**	**89.8**	**48.5**	**37.1**
$K = 8$	35.0	57.1	89.6	47.5	36.0

(a) Mode-level Prototype Number

Coefficient of SPL	Recall(%) Tri.↑	Obj.↑	Pred.↑	mRecall(%) Obj.↑	Pred.↑
$\mu = 0$	33.7	56.5	88.1	45.2	35.2
$\mu = 0.9$	35.0	57.5	89.5	47.3	35.3
$\mu = 0.99$	**35.8**	**58.2**	**89.8**	**48.5**	**37.1**
$\mu = 0.999$	34.8	57.2	89.6	48.4	35.5

(b) Semantic-level Prototype Updating Coefficient μ

Learning Node	Branch Edge	Recall(%) Tri.↑	Obj.↑	Pred.↑	mRecall(%) Obj.↑	Pred.↑
		32.9	54.5	88.7	45.4	35.2
✓		34.0	57.2	88.4	47.2	35.2
	✓	34.8	55.1	89.2	46.0	37.0
✓	✓	**35.8**	**58.2**	**89.8**	**48.5**	**37.1**

(c) Prototype Learning Branch

Clustering Strategy	Recall(%) Tri.↑	Obj.↑	Pred.↑	mRecall(%) Obj.↑	Pred.↑
None	32.9	54.5	88.7	45.4	35.2
Contrastive loss	34.4	56.9	**90.2**	44.1	35.0
Center loss	33.6	55.9	88.1	43.9	34.7
Ours	**35.8**	**58.2**	89.8	**48.5**	**37.1**

(d) Mode-level Prototype Learning

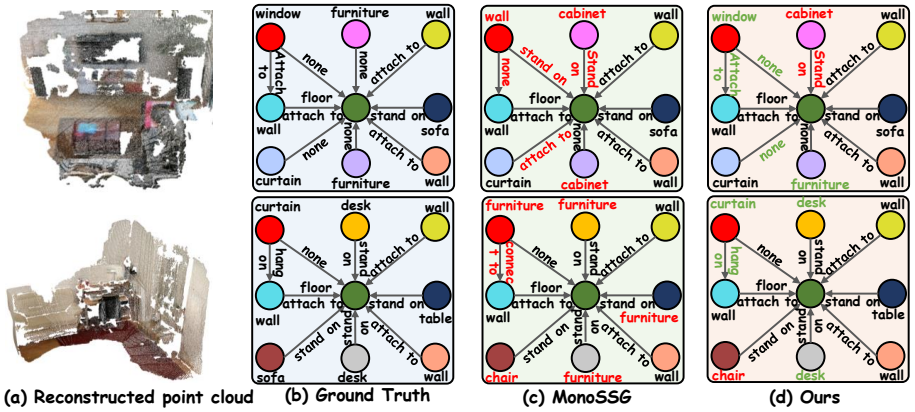

Figure 4: Qualitative results on the 3DSSG [55] dataset. Red edge: misclassified edges from MonoSSG, green edge: edges corrected by our method, red node: misclassified node.

Mode-level Prototype Learning Objective. Table 4d compares $\mathcal{L}_{\text{SDL}}$ with contrastive and center losses. Our objective improves Obj. from 54.5% to **58.2**% and mRecall from 45.4% to **48.5**%, demonstrating the effectiveness of scatter-matrix-based feature-prototype modeling. Furthermore, we visualize the feature distributions to investigate how different loss functions affect the structure of the feature space. As shown in Fig. 3, our method achieves more compact intra-class clusters and clearer inter-class separation than other objectives in feature space.

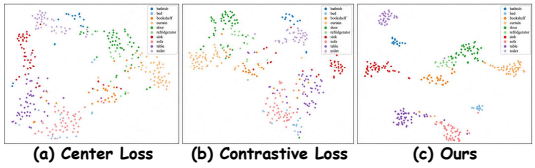

Figure 3: The t-SNE visualization of feature distributions under different Mode-level Prototype Learning.

4.4 Quality Analysis

We present the qualitative comparison in Fig. 4. Compared with MonoSSG, our method generates scene graphs that are more consistent with the ground truth. It better separates similar node categories, such as **wall**, **curtain**, and **cabinet**, while more reliably predicting relations such as **attached to** and **standing on**. In addition, our method reduces incorrect nodes and invalid edges, leading to more accurate and compact scene graph structures. More results are provided in §D of Appendix.

5 Conclusion

In this paper, we propose HPSSG, a human-inspired hierarchical prototype learning framework for RGB-D based 3D scene graph generation. HPSSG models nodes and edges with semantic-level and mode-level prototypes, enabling the model to preserve stable category semantics while capturing diverse latent patterns within each category. Hierarchy-aware Prototype Entailment Learning further maintains hierarchical consistency, and Prior-guided Compositional Prototype Learning incorporates subject-object pair semantics into edge modeling. Experiments on the 3DSSG demonstrate that our method outperforms various leading methods without any additional inference cost.

AI USE STATEMENT

We used generative AI tools for language proofreading and to improve the clarity of the manuscript. We did not use generative AI tools to generate experimental results, perform data analysis, or draw scientific conclusions. In addition, generative AI tools were used to assist with parts of the coding process and to check for grammatical errors. All content generated or modified with the assistance of AI was carefully reviewed and verified by the authors. We take full responsibility for the final content of this manuscript.

REFERENCES

[1] Jiawei Fu, Tiantian Zhang, Kai Chen, and Qi Dou. Hybrid reciprocal transformer with triplet feature alignment for scene graph generation. In *CVPR*, pages 8953–8963, 2025. 1, 14

[2] Wenhuan Huang, Yi JI, Guiqian Zhu, Li Ying, and Chunping Liu. Navigating the unseen: Zero-shot scene graph generation via capsule-based equivariant features. In *CVPR*, pages 29448–29457, 2025. 1, 14

[3] Mu Chen, Liulei Li, Wenguan Wang, and Yi Yang. Diffvsgg: Diffusion-driven online video scene graph generation. In *CVPR*, pages 29161–29172, 2025. 1, 14

[4] Yanjun Li, Zhaoyang Li, Honghui Chen, and Lizhi Xu. Unbiased video scene graph generation via visual and semantic dual debiasing. In *CVPR*, pages 19047–19056, 2025. 1, 14

[5] Shun-Cheng Wu, Keisuke Tateno, Nassir Navab, and Federico Tombari. Incremental 3d semantic scene graph prediction from rgb sequences. In *CVPR*, pages 5064–5074, 2023. 1, 2, 3, 7, 8, 14, 18

[6] Chaoyi Zhang, Xitong Yang, Ji Hou, Kris Kitani, Weidong Cai, and Fu-Jen Chu. Egosg: Learning 3d scene graphs from egocentric rgb-d sequences. In *CVPR*, pages 2535–2545, 2024. 1, 3, 8

[7] Abdelrhman Werby, Chenguang Huang, Martin Büchner, Abhinav Valada, and Wolfram Burgard. Hierarchical open-vocabulary 3d scene graphs for language-grounded robot navigation. In *ICRA*, 2024. 1

[8] Hang Yin, Xiuwei Xu, Zhenyu Wu, Jie Zhou, and Jiwen Lu. Sg-nav: Online 3d scene graph prompting for llm-based zero-shot object navigation. In *NeurIPS*, volume 37, pages 5285–5307, 2024. 1

[9] Muhammad Shaheer, Jose Andres Millan-Romera, Hriday Bavle, Jose Luis Sanchez-Lopez, Javier Civera, and Holger Voos. Graph-based global robot localization informing situational graphs with architectural graphs. In *IROS*, pages 9155–9162. IEEE, 2023. 1

[10] Yang Miao, Francis Engelmann, Olga Vysotska, Federico Tombari, Marc Pollefeys, and Dániel Béla Baráth. Scenegraphloc: Cross-modal coarse visual localization on 3d scene graphs. In *ECCV*, pages 127–150. Springer, 2024. 1

[11] Tatiana Zemskova and Dmitry Yudin. 3dgraphllm: Combining semantic graphs and large language models for 3d scene understanding. In *ICCV*, pages 8885–8895. IEEE, 2025. 1

[12] Yu Fei, Quan Deng, Shengeng Tang, Li Yuehua, and Lechao Cheng. Open-world 3d scene graph generation for retrieval-augmented reasoning. In *AAAI*, volume 40, pages 3894–3902, 2026. 1

[13] Yuheng Liu, Xinke Li, Yuning Zhang, Lu Qi, Xin Li, Wenping Wang, Chongshou Li, Xueting Li, and Ming-Hsuan Yang. Controllable 3d outdoor scene generation via scene graphs. *arXiv preprint arXiv:2503.07152*, 2025. 1

[14] Zhide Zhong, Jiakai Cao, Songen Gu, Sirui Xie, Liyi Luo, Hao Zhao, Guyue Zhou, Haoang Li, and Zike Yan. Structured-nerf: Hierarchical scene graph with neural representation. In *ECCV*, pages 184–201. Springer, 2024. 1

[15] Changsheng Lv, Mengshi Qi, Xia Li, Zhengyuan Yang, and Huadong Ma. Sgformer: Semantic graph transformer for point cloud-based 3d scene graph generation. In *AAAI*, volume 38, pages 4035–4043, 2024. 1

[16] KunHo Heo, GiHyeon Kim, SuYeon Kim, and MyeongAh Cho. Object-centric representation learning for enhanced 3d semantic scene graph prediction. In *NeurIPS*, 2025. 1, 3, 7, 8, 18

[17] Mingtao Feng, Chenbo Yan, Zijie Wu, Weisheng Dong, Yaonan Wang, and Ajmal Mian. Hyperrectangle embedding for debiased 3d scene graph prediction from rgb sequences. *IEEE TPAMI*, 2025. 2, 3, 7, 8, 14

[18] Qi Xun Yeo, Yanyan Li, and Gim Hee Lee. Statistical confidence rescoring for robust 3d scene graph generation from multi-view images. In *ICCV*, pages 24999–25008, 2025. 2, 3, 7, 14, 18

[19] Mingtao Feng, Chenbo Yan, Zijie Wu, Weisheng Dong, Yaonan Wang, and Ajmal Mian. History-enhanced 3d scene graph reasoning from rgb-d sequences. *IEEE TCSVT*, 2025. 2, 3, 8

[20] Chaoyi Zhang, Jianhui Yu, Yang Song, and Weidong Cai. Exploiting edge-oriented reasoning for 3d point-based scene graph analysis. In *CVPR*, pages 9705–9715, 2021. 3

[21] Shoulong Zhang, Aimin Hao, Hong Qin, et al. Knowledge-inspired 3d scene graph prediction in point cloud. In *NeurIPS*, volume 34, pages 18620–18632, 2021. 3

[22] Ziqin Wang, Bowen Cheng, Lichen Zhao, Dong Xu, Yang Tang, and Lu Sheng. Vl-sat: Visual-linguistic semantics assisted training for 3d semantic scene graph prediction in point cloud. In *CVPR*, pages 21560–21569, 2023. 3, 8

[23] Sebastian Koch, Pedro Hermosilla, Narunas Vaskevicius, Mirco Colosi, and Timo Ropinski. Lang3dsg: Language-based contrastive pre-training for 3d scene graph prediction. In *3DV*, pages 1037–1047. IEEE, 2024. 3

[24] Qixiang Ma, Runze Fan, Lizhi Zhao, Jian Wu, Sio-Kei Im, and Lili Wang. Sgsg: Stroke-guided scene graph generation. *IEEE TVCG*, 2025. 3

[25] Yucheng Huang, Luping Ji, Ruijie Xiao, and Jiayuan Sun. Multi-view invariance learning for 3d scene graph pre-training via collaborative cross-modal regularization. In *AAAI*, pages 5203–5211, 2026. 3

[26] Yanni Ma, Hao Liu, Yulan Guo, Theo Gevers, and Martin R Oswald. Edge-centric relational reasoning for 3d scene graph prediction. In *AAAI*, pages 7847–7855, 2026. 3

[27] Mingtao Feng, Haoran Hou, Liang Zhang, Yulan Guo, Hongshan Yu, Yaonan Wang, and Ajmal Mian. Exploring hierarchical spatial layout cues for 3d point cloud based scene graph prediction. *IEEE TMM*, 27:731–743, 2023. 3

[28] Mingtao Feng, Haoran Hou, Liang Zhang, Zijie Wu, Yulan Guo, and Ajmal Mian. 3d spatial multimodal knowledge accumulation for scene graph prediction in point cloud. In *CVPR*, pages 9182–9191, 2023. 3

[29] Sebastian Koch, Narunas Vaskevicius, Mirco Colosi, Pedro Hermosilla, and Timo Ropinski. Open3dsg: Open-vocabulary 3d scene graphs from point clouds with queryable objects and open-set relationships. In *CVPR*, pages 14183–14193, 2024. 3

[30] Lianggangxu Chen, Xuejiao Wang, Jiale Lu, Shaohui Lin, Changbo Wang, and Gaoqi He. Clip-driven open-vocabulary 3d scene graph generation via cross-modality contrastive learning. In *CVPR*, pages 27863–27873, 2024. 3

[31] Chenyangguang Zhang, Alexandros Delitzas, Fangjinhua Wang, Ruida Zhang, Xiangyang Ji, Marc Pollefeys, and Francis Engelmann. Open-vocabulary functional 3d scene graphs for real-world indoor spaces. In *CVPR*, pages 19401–19413, 2025. 3

[32] Fei Yu, Quan Deng, Shengeng Tang, Yuehua Li, and Lechao Cheng. Open-world 3d scene graph generation for retrieval-augmented reasoning. *arXiv preprint arXiv:2511.05894*, 2025. 3

[33] Shun-Cheng Wu, Johanna Wald, Keisuke Tateno, Nassir Navab, and Federico Tombari. Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences. In *CVPR*, pages 7515–7525, 2021. 3, 7, 8, 18

[34] Hao-Yu Hou, Chun-Yi Lee, Motoharu Sonogashira, and Yasutomo Kawanishi. Fross: Faster-than-real-time online 3d semantic scene graph generation from rgb-d images. In *ICCV*, pages 28818–28827, 2025. 3

[35] Chiyu Ma, Jon Donnelly, Wenjun Liu, Soroush Vosoughi, Cynthia Rudin, and Chaofan Chen. Interpretable image classification with adaptive prototype-based vision transformers. In *NeurIPS*, pages 41447–41493, 2024. 3

[36] Hongyu Qu, Jianan Wei, Xiangbo Shu, and Wenguan Wang. Learning clustering-based prototypes for compositional zero-shot learning. *arXiv preprint arXiv:2502.06501*, 2025. 3

[37] Siyuan Li, Zedong Wang, Zicheng Liu, Cheng Tan, Haitao Lin, Di Wu, Zhiyuan Chen, Jiangbin Zheng, and Stan Z Li. Moganet: Multi-order gated aggregation network. In *ICLR*, volume 2024, pages 37393–37427, 2024. 3

[38] Wenhai Wang, Jifeng Dai, Zhe Chen, Zhenhang Huang, Zhiqi Li, Xizhou Zhu, Xiaowei Hu, Tong Lu, Lewei Lu, Hongsheng Li, et al. Internimage: Exploring large-scale vision foundation models with deformable convolutions. In *CVPR*, pages 14408–14419, 2023. 3

[39] Zhijie Zhu, Lei Fan, Maurice Pagnucco, and Yang Song. Interpretable image classification via non-parametric part prototype learning. In *CVPR*, pages 9762–9771, 2025. 3

[40] Jian Wang, Tianhong Dai, Bingfeng Zhang, Siyue Yu, Eng Gee Lim, and Jimin Xiao. Pot: Prototypical optimal transport for weakly supervised semantic segmentation. In *CVPR*, pages 15055–15064, 2025. 3

[41] Jinhong Lin, Cheng-En Wu, Huanran Li, Jifan Zhang, Yu Hen Hu, and Pedro Morgado. From prototypes to general distributions: An efficient curriculum for masked image modeling. In *CVPR*, pages 20028–20038, 2025. 3

[42] Weiliang Tang, Biqi Yang, Xianzhi Li, Yun-Hui Liu, Pheng-Ann Heng, and Chi-Wing Fu. Prototypical variational autoencoder for 3d few-shot object detection. pages 2566–2579, 2023. 3

[43] Haodong Lu, Dong Gong, Shuo Wang, Jason Xue, Lina Yao, and Kristen Moore. Learning with mixture of prototypes for out-of-distribution detection. *arXiv preprint arXiv:2402.02653*, 2024. 3

[44] Xi Wang, Xu Yang, Jie Yin, Kun Wei, and Cheng Deng. Long-tail class incremental learning via independent sub-prototype construction. In *CVPR*, pages 28598–28607, 2024. 3

[45] Hao Zhu and Piotr Koniusz. Transductive few-shot learning with prototype-based label propagation by iterative graph refinement. In *CVPR*, pages 23996–24006, 2023. 3

[46] Dongyao Jiang, Hui Chen, Haodong Jing, Yongqiang Ma, and Nanning Zheng. Mrsp: Learn multi-representations of single primitive for compositional zero-shot learning. In *ECCV*, pages 1–17. Springer, 2024. 3

[47] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, pages 652–660, 2017. 3, 16

[48] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 3, 16

[49] Xiao Cui, Yulei Qin, Wengang Zhou, Hongsheng Li, and Houqiang Li. Optical: Leveraging optimal transport for contribution allocation in dataset distillation. In *CVPR*, pages 15245–15254, 2025. 5

[50] Liangliang Shi, Yufeng Li, Kaipeng Zeng, Yihui Tu, and Junchi Yan. Optimal flow transport and its entropic regularization: a gpu-friendly matrix iterative algorithm for flow balance satisfaction. In *ICLR*, 2025. 5

[51] Gabriel Y Arteaga, Marius Aasan, Rwiddhi Chakraborty, Martine Hjelkrem-Tan, Thalles Silva, Michael Kampffmeyer, and Adín Ramírez Rivera. Why prototypes collapse: Diagnosing and preventing partial collapse in prototypical self-supervised learning. *arXiv preprint arXiv:2510.20108*, 2025. 5

[52] Tuo Chen, Jie Gui, Minjing Dong, Ju Jia, Lanting Fang, and Jian Liu. Backdooring self-supervised contrastive learning by noisy alignment. In *ICCV*, pages 3684–3693, 2025. 5

[53] Haowen Li, Zhenfeng Fan, Zhang Wen, Zhengzhou Zhu, and Yunjin Li. Aicomposer: Any style and content image composition via feature integration. In *ICCV*, pages 16840–16850, 2025. 6

[54] Ran Eisenberg, Jonathan Svirsky, and Ofir Lindenbaum. Coper: Correlation-based permutations for multi-view clustering. In *ICLR*, 2025. 6

[55] Johanna Wald, Helisa Dhamo, Nassir Navab, and Federico Tombari. Learning 3d semantic scene graphs from 3d indoor reconstructions. In *CVPR*, pages 3961–3970, 2020. 7, 8, 9, 14, 17, 18, 19

[56] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *CVPR*, pages 5410–5419, 2017. 7, 8, 18

[57] Paul Gay, James Stuart, and Alessio Del Bue. Visual graphs from motion (vgfm): Scene understanding with object geometry reasoning. In *ACCV*, pages 330–346. Springer, 2018. 7, 8, 18

[58] Johanna Wald, Armen Avetisyan, Nassir Navab, Federico Tombari, and Matthias Nießner. Rio: 3d object instance re-localization in changing indoor environments. In *ICCV*, pages 7658–7667, 2019. 7, 14

[59] Kyunghyun Cho, Bart Van Merriënboer, Çağlar Gulçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014. 16

This appendix provides additional details for the ICLR 2027 submission, titled *"Hierarchical Prototype Learning for 3D Scene Graph Generation from RGB-D Sequences"*. The appendix is organized as follows:

- §A Detailed Experimental Setup.
- §B More Details about Feature Initialization
- §C More Quantitative Results.
- §D More Qualitative Visualization.
- §E Discussion.

A Detailed Experimental Setup

A.1 Dataset Description

Built upon the real-world 3RScan benchmark [58], the 3DSSG dataset [55] is, to our knowledge, the only publicly available dataset for 3D scene graph generation [17, 18]. It comprises 1335 reconstructed indoor scenes represented as 3D point clouds with instance-level annotations, associated with approximately 363k RGB-D frames. Notably, this dataset poses significant challenges: as the images were acquired using consumer-level devices, the sequences typically suffer from low frame rates (10 Hz), motion blur, and image jitter caused by sudden camera movements. Furthermore, each scene is essentially composed of multiple short video clips with varying motion trajectories spliced together; this discontinuity and the low quality of the visual data further exacerbate the difficulty of the task. In terms of scale, the dataset encompasses 160 node categories and 26 edge categories, containing a total of 48K nodes and 544K edges. Following the standard split, we partition the dataset into 1061 training scans, 117 validation scans, and 157 test scans.

A.2 Evaluation

Evaluation Tasks. (1) **Object Classification:** This task aims to assign the correct semantic label to each node within the scene graph, such as identifying a "chair" or "table". It evaluates the accuracy of entity recognition alone, independent of the prediction results for the connections between nodes. (2) **Predicate Classification:** This task focuses on predicting the semantic relationships embedded within the edges connecting node pairs, such as distinguishing between "standing on" and "supported by". It primarily evaluates the model's capability to reason about object interactions, independent of the classification accuracy of the object labels themselves. (3) **Triplet Classification:** This task involves the joint prediction of the triplet and serves as the most rigorous metric for holistic scene understanding. Under this criterion, a prediction is deemed correct only if the subject class, object class, and the connecting relationship predicate all fully match the ground truth.

Evaluation Metrics. For a fair comparison, we follow the protocol established by Wu et al. [5], reporting results both with and without the 'none' category for a comprehensive evaluation. Notably, our *recall@1* metric differs from the standard *recall@N* typically used in 2D scene graph tasks [1, 2, 3, 4]. While the standard approach selects the top N triplets from a global ranking, our metric evaluates the accuracy of the single most confident prediction (subject, predicate, object) for each detected triplet. Given that our model produces an unconstrained number of triplets, this metric serves as a practical proxy for *recall@*∞ under graph constraints. For simplicity, we refer to this metric as *recall* throughout the paper.

A.3 Training Details

In terms of network configuration, we set PointNet to be trainable, while freezing the parameters of DINOv2 to directly utilize its pre-trained weights without fine-tuning. For multi-view image inputs, all images are resized to a resolution of 224×224 and subsequently fed into DINOv2 for feature extraction. If the validation metric does not improve for 10 consecutive epochs, the learning rate is multiplied by 0.9. Regarding the number of Gaussian components, we set it to 5. The momentum coefficient μ for semantic-level prototype updating is set to 0.99 (Eq. 2). In mode-level prototype initialization, $\sigma = 0.05$ (Eq. 4). In scatter-guided discriminative learning, $\gamma = 0.01$ (Eq. 10). The

Sinkhorn-Knopp algorithm is iterated for 5 steps. In hierarchy-aware prototype entailment learning, τ_n is set to 0.65 rad (Eq. 11). For the coefficient λ_1 in the loss function (Eq. 13), we adopt a dynamic weighting strategy. Specifically, λ_1 is set to 1 for node prediction. For edge prediction, considering the imbalance between the numbers of nodes and edges in different scenes, λ_1 is dynamically computed based on the node-edge ratio: $\lambda_1 = N_{edge}/N_{node}$, where N_{node} and N_{edge} denote the numbers of nodes and edges in the current scene, respectively.

B More Details about Our Method

Algorithm 1 Pattern-aware Prototype Assignment

Require: Feature embeddings $\mathbf{Z}^t = \{\mathbf{z}_i^t\}_{i=1}^N$; labels $\mathbf{y}^t$; fine prototypes $\mathbf{P}^t \in \mathbb{R}^{C_t \times K_t \times d}$; class-wise feature queue $\mathcal{Q}^t$; OT temperature ϵ; Sinkhorn iterations T_s, where $t \in \{n, e\}$

Ensure: Feature-prototype assignment matrix $\mathbf{A}^t \in \mathbb{R}^{N \times K_t}$

Normalize features and prototypes: $\bar{\mathbf{Z}}^t \leftarrow \ell_2(\mathbf{Z}^t)$, $\bar{\mathbf{P}}^t \leftarrow \ell_2(\mathbf{P}^t)$
Initialize $\mathbf{A}^t \leftarrow \mathbf{0}$
for each class $c = 1, \ldots, C_t$ **do**
  Select current features of class c: $\mathcal{I}_c \leftarrow \{i \mid y_i^t = c\}$
  if $|\mathcal{I}_c| = 0$ **then**
    continue
  end if
  $\mathbf{X}_c \leftarrow \{\bar{\mathbf{z}}_i^t \mid i \in \mathcal{I}_c\}$
  Retrieve historical features $\mathbf{H}_c \leftarrow \mathcal{Q}_c^t$
  Construct augmented class features: $\tilde{\mathbf{X}}_c \leftarrow \mathrm{Concat}(\mathbf{X}_c, \mathbf{H}_c)$
  Let $N_c^+ = |\tilde{\mathbf{X}}_c|$ and $\mathbf{P}_c^t \in \mathbb{R}^{K_t \times d}$ be the prototypes of class c
  Compute feature-prototype affinity:

$$\mathbf{S}_c = \frac{\mathbf{P}_c^t \tilde{\mathbf{X}}_c^\top}{\epsilon} \in \mathbb{R}^{K_t \times N_c^+}$$

  Initialize transport matrix:

$$\mathbf{M}_c \leftarrow \exp(\mathbf{S}_c - \max(\mathbf{S}_c))$$

  Normalize total mass: $\mathbf{M}_c \leftarrow \mathbf{M}_c / \sum_{k,j} \mathbf{M}_c(k, j)$
  for $s = 1, \ldots, T_s$ **do**
    Prototype equipartition:

$$\mathbf{M}_c \leftarrow \frac{\mathbf{M}_c}{K_t \sum_j \mathbf{M}_c(:, j)}$$

    Feature unique-assignment constraint:

$$\mathbf{M}_c \leftarrow \frac{\mathbf{M}_c}{N_c^+ \sum_k \mathbf{M}_c(k, :)}$$

  end for
  Rescale transport so that each feature has unit assignment mass:

$$\mathbf{M}_c \leftarrow N_c^+ \mathbf{M}_c$$

  Keep assignments for current-batch features only:

$$\mathbf{A}_{\mathcal{I}_c}^t \leftarrow \mathbf{M}_c[:, 1 : |\mathcal{I}_c|]^\top$$

end for
Update class-wise queue $\mathcal{Q}^t$ with $\bar{\mathbf{Z}}^t$ and $\mathbf{y}^t$
return $\mathbf{A}^t$

B.1 More Details about Feature Initialization

After obtaining the reconstructed 3D point cloud and multi-view RGB images, we first construct object-level node and edge representations from both visual and geometric cues. For each object

node i, we aggregate its multi-view observations to obtain the visual representation $\boldsymbol{e}_i$:

$$\boldsymbol{e}_i = \frac{1}{|\mathcal{K}_i|} \sum_{k \in \mathcal{K}_i} \mathtt{E}(\boldsymbol{I}_{i,k}), \tag{14}$$

where $\boldsymbol{I}_{i,k}$ denotes the image observation of node i from the k-th view, $\mathcal{K}_i$ is the set of available view indices for node i, and $\mathtt{E}(\cdot)$ denotes the DINOv2 [48] image encoder. Meanwhile, we use PointNet [47] to extract the geometric representation $\boldsymbol{g}_i$ from the corresponding 3D points of node i. The visual and geometric features are then fused to form the initial node feature $\hat{\boldsymbol{v}}_i$:

$$\hat{\boldsymbol{v}}_i = \mathtt{MLP}\left([\boldsymbol{e}_i, \boldsymbol{g}_i]\right). \tag{15}$$

For each directed edge $i \to j$, we encode its spatial configuration according to the relative geometric attributes between the subject node i and the object node j. Specifically, the initial edge feature $\hat{\boldsymbol{y}}_{i \to j}$ is computed as:

$$\hat{\boldsymbol{y}}_{i \to j} = \mathtt{MLP}\left([\boldsymbol{c}_j - \boldsymbol{c}_i,\ \boldsymbol{b}_j - \boldsymbol{b}_i,\ \boldsymbol{r}_{i \to j}]\right), \tag{16}$$

where $\boldsymbol{c}_i$ and $\boldsymbol{c}_j$ denote the center coordinates of nodes i and j, $\boldsymbol{b}_i$ and $\boldsymbol{b}_j$ are their bounding-box parameter vectors, and $\boldsymbol{r}_{i \to j}$ represents the relative pose descriptor, which encodes the relative orientation between the two nodes.

Based on the initialized node and edge features, we further perform graph message passing to capture contextual interactions among objects and relations. At each iteration t, the feature-wise attention network $\mathtt{FAN}(\cdot)$ is used to aggregate relation-aware messages from neighboring nodes. The node feature $\boldsymbol{v}_i^t$ is updated by Gated Recurrent Unit (GRU) [59] as:

$$\boldsymbol{v}_i^t = \mathtt{GRU}\left(\boldsymbol{v}_i^{t-1}, \mathtt{MLP}\left(\left[\hat{\boldsymbol{v}}_i,\ \max_{j \in \mathcal{F}(i)} \mathtt{FAN}\left(\hat{\boldsymbol{v}}_i, \boldsymbol{e}_{i \to j}, \hat{\boldsymbol{v}}_j\right)\right]\right)\right), \tag{17}$$

where $\mathcal{F}(i)$ denotes the set of neighboring nodes of node i. Similarly, the edge representation $\boldsymbol{y}_{i \to j}^t$ is updated by incorporating the features of its two endpoint nodes and their geometric relation:

$$\boldsymbol{y}_{i \to j}^t = \mathtt{GRU}\left(\boldsymbol{z}_{i \to j}^{t-1}, \mathtt{MLP}\left([\hat{\boldsymbol{v}}_i,\ \hat{\boldsymbol{y}}_{i \to j},\ \hat{\boldsymbol{v}}_j]\right)\right). \tag{18}$$

After several rounds of message passing, we obtain the node representation $\boldsymbol{v}_i$ and edge representation $\boldsymbol{y}_{i \to j}$, which are used for subsequent node and edge prediction.

B.2 More Details about Pattern-aware Prototype Assignment

For the pattern-aware prototype assignment in Mode-level Prototype Learning, we introduce a feature–prototype assignment strategy based on the Sinkhorn-Knopp optimal transport algorithm. The detailed optimization procedure is presented in Algorithm 1. The goal of this process is to assign category-specific features to multiple mode prototypes, enabling each prototype to capture a distinct latent pattern within the same category.

Feature Collection and Augmentation. Specifically, for each category c, we first collect all feature embeddings belonging to this category from the current mini-batch. Since the number of instances for each category may be limited in a single batch, directly performing prototype assignment may lead to unstable optimization and incomplete pattern discovery. To alleviate this issue, we maintain a class-wise feature queue that stores historical features from previous iterations. The current features are then combined with the retrieved historical features to construct an augmented feature set:

$$\mathcal{X}_c = \mathrm{Concat}(\mathcal{X}_c^t, \mathcal{H}_c), \tag{19}$$

where $\mathcal{X}_c^t$ denotes the current batch features and $\mathcal{H}_c$ represents historical features from the queue. This augmentation provides richer feature diversity and facilitates more stable mode prototype learning.

Feature-Prototype Affinity Computation. Given the augmented features and the corresponding K_c mode prototypes of category c, we first normalize both feature embeddings and prototypes. Then, the feature–prototype affinity matrix is computed by:

$$S_c = \frac{P_c^t X_c^\top}{\epsilon}, \tag{20}$$

where P_c^t denotes the mode prototypes of category c, and ϵ is the temperature coefficient. Each element in S_c measures the compatibility between a feature instance and a mode prototype, providing the basis for subsequent prototype assignment.

Table 5: Per-class performance comparison of 3DSGG methods for object recall (%) on 3DSSG [55], which from ground-truth point cloud. The best and the second best results are highlighted in bold and underlined.

Method	bath.	bed	bksh.	cab.	chair	cntr.	curt.	desk	door	floor	ofur.	pic.	refri.	show.	sink	sofa	table	toil	wall	wind	mean
IMP	25.0	100	28.6	47.3	67.9	64.5	74.8	**58.3**	47.7	96.3	17.6	58.8	11.1	14.3	53.3	65.8	61.2	92.6	83.8	54.2	56.2
VGfM	75.0	100	14.3	51.9	88.7	71.0	70.3	12.5	61.7	98.2	24.5	67.1	0	14.3	71.7	71.1	72.7	88.9	85.0	51.8	59.5
SGPN	50.0	33.3	0	48.1	80.4	58.1	36.9	45.8	67.3	95.1	21.3	17.6	33.3	42.9	61.7	60.5	53.9	59.3	57.4	31.3	47.7
SGFN	50.0	33.3	0	44.2	75.4	**83.9**	56.8	50.0	60.7	96.9	36.2	82.4	11.1	28.6	80.0	40.8	55.2	63.0	72.0	51.8	53.6
MonoSSG†	100	100	42.9	62.4	92.9	80.6	89.2	33.3	81.3	**98.2**	57.4	96.5	44.4	100	90.0	78.9	73.9	100	82.6	59.0	78.2
MonoSSG	100	100	**57.1**	62.4	**92.9**	83.9	88.3	33.3	**83.2**	97.5	59.0	**100**	44.4	100	93.3	**80.3**	73.9	100	82.9	61.4	79.7
Ours	**100**	**100**	42.9	**76.7**	87.9	80.6	**90.1**	45.8	81.3	97.5	**68.1**	95.3	**66.7**	**100**	**95.0**	76.3	**86.1**	**100**	**89.3**	**78.3**	**82.9**

Table 6: Per-class performance comparison of 3DSGG methods for predicate recall (%) on 3DSSG [55], which from ground-truth point cloud. The best and the second best results are highlighted in bold and underlined.

Method	attached to	build in	connected to	hanging on	none	part of	standing on	supported by	mean
IMP	58.1	30.8	21.7	29.1	97.7	25.0	71.9	0	41.8
VGfM	63.1	33.3	28.3	25.4	**98.1**	43.8	81.5	1.1	46.8
SGPN	72.8	74.4	34.8	53.0	97.7	68.8	84.0	6.6	61.5
SGFN	68.7	66.7	56.5	57.5	96.7	75.0	81.8	2.2	63.1
MonoSSG†	74.3	87.2	65.2	56.0	97.4	37.5	**92.1**	8.8	64.8
MonoSSG	**82.2**	87.2	**73.9**	65.7	97.6	75.0	91.9	**9.9**	72.9
Ours	79.0	**92.3**	60.9	**73.9**	97.6	**100.0**	90.6	3.3	**74.7**

Table 7: Efficiency comparison on 3DSSG [55]. Here, we report training memory and training time per epoch, and inference parameters and speed for each model. See in §C for more details.

Method	Inference Params↓	Training Memory↓	Training Time↓	Inference Speed↓
Baseline	312.5M	1.54G	0.68s	0.49s
HPSSG (Ours)	312.5M	1.62G	0.72s	0.49s

Balanced Prototype Assignment with Sinkhorn Optimization. Instead of assigning each feature to the most similar prototype independently, we formulate the assignment process as an optimal transport problem. The initial transport matrix is obtained by transforming the affinity matrix, and the Sinkhorn-Knopp algorithm is adopted to iteratively optimize the assignment matrix. Specifically, the algorithm alternately normalizes the rows and columns of the transport matrix, enforcing two constraints: (1) each prototype receives balanced assignment mass, preventing a single dominant prototype from capturing most samples; (2) each feature is assigned with a unique distribution over prototypes. Through this balanced assignment mechanism, different mode prototypes are encouraged to capture diverse semantic patterns within the same category.

Prototype Updating. After obtaining the optimized assignment matrix, only the assignments corresponding to the current batch features are retained for prototype updating. The resulting assignment weights are then used to update the mode prototypes through the proposed prototype learning strategy. As a result, multiple prototypes within each category can progressively adapt to different intra-class variations, such as diverse object appearances, geometric configurations, and relation patterns.

Overall, the proposed pattern-aware prototype assignment enables category-specific mode prototypes to discover meaningful latent patterns automatically, while avoiding the collapse problem where dominant patterns overwhelm other prototypes.

C More Quantitative Results

Per-class Performance Comparison with Existing 3DSGG Methods. Tables 5 and 6 present per-class object and predicate recall on the 3DSSG dataset using ground-truth point clouds, respectively. As shown in Table 5, our method achieves the highest mean recall of **82.9**% for object prediction, surpassing MonoSSG by **3.2**%. At the category level, our method obtains the best results on cabinet, curtain, otherfurniture, refrigerator, sink, table, wall, and window. In particular, compared with MonoSSG, the recall of cabinet increases from 62.4% to **76.7**%, while table and window improve from 73.9% and 61.4% to **86.1**% and **78.3**%, respectively. These results demonstrate the effectiveness of our method in improving recognition across diverse object categories.

For predicate prediction, Table 6 shows that our method achieves the highest mean recall of **74.7**%, outperforming MonoSSG by **1.8**%. Specifically, our method obtains the best results on build in,

Table 8: **Additional Quantitative results** (§C) are evaluated on 3DSSG [55] using ground-truth point clouds within 160 node and 26 edge classes. The best and the second best results are highlighted in bold and underlined. $*$: without *none* relationship.

Method	Input	Recall(%)						mRecall(%)	
		Tri.↑	Tri.*↑	Obj.↑	Obj.*↑	Pred.↑	Pred.*↑	Obj.↑	Pred.↑
IMP [56] [CVPR2017]	GT	64.1	4.9	42.9	42.8	16.2	16.3	16.0	3.6
VGfM [57] [ACCV2018]	GT	64.5	5.9	45.9	46.0	17.4	17.6	19.1	5.5
SGPN [55] [CVPR2020]	GT	64.8	6.8	27.9	27.8	**67.0**	**67.0**	12.1	20.9
SGFN [33] [CVPR2021]	GT	64.6	6.6	36.8	36.7	48.4	48.4	16.2	14.4
MonoSSG [5] [CVPR2023]	GT	67.5	14.7	<u>53.3</u>	<u>53.2</u>	48.0	48.0	<u>28.9</u>	<u>24.7</u>
OCRL [16] [NeurIPS2025]	GT	<u>67.6</u>	<u>15.6</u>	41.5	41.5	42.6	42.6	18.8	22.7
SCRSSG [18] [ICCV2025]	GT	57.7	-	32.1	32.1	10.9	10.9	19.6	2.3
HPSSG (Ours)	GT	**69.4**	**20.0**	**62.7**	**62.7**	<u>52.6</u>	<u>52.6</u>	**36.5**	**30.7**

hanging on, and part of. Compared with MonoSSG, the recall of build in increases from 87.2% to **92.3**%, hanging on improves from 65.7% to **73.9**%, and part of rises from 75.0% to **100**%. Our method also achieves the second-best recall of 79.0% on attached to. These results demonstrate improved category-balanced predicate recognition and support the effectiveness of the proposed framework in learning discriminative representations for both object nodes and relational edges.

Efficiency Analysis. For a fair comparison, we evaluate training-time memory consumption and per-epoch training time on Scene 0958220d-e2c2-2de1-9710-c37018da1883. For inference-related metrics, including inference speed and inference parameters, we report results on Scene d7d40d4e-7a5d-2b36-97e7-34324c52ac42. As shown in Table 7, HPSSG introduces only marginal training overhead compared with the baseline. Specifically, the training memory slightly increases from 1.54G to 1.62G, and the training time per epoch increases from 0.68s to 0.72s. Meanwhile, HPSSG keeps the same inference parameters as the baseline, *i.e.*, 312.5M, and achieves identical inference speed of 0.49s. These results demonstrate that the proposed hierarchical prototype learning framework improves representation learning with negligible computational overhead, especially without increasing the inference cost.

More comparison experiments in larger label space. We further extend the original setting from 20 object categories and 8 relation categories to 160 object categories and 26 relation categories, providing a more challenging evaluation under an expanded label space. As shown in Table 8, Our method achieves the best performance on both Tri. and Tri.*, reaching **69.4**% and **20.0**%, respectively. Moreover, it obtains **36.5**% (**+7.6**%) and **30.7**% (**+6.0**%) on Obj. and Pred. under mRecall. These results demonstrate that our method remains effective in a more challenging label space while introducing no additional inference overhead.

D More Qualitative Visualization

We provide additional visual comparisons between our method GMPSSG and MonoSSG on **larger-scale scenes** from the 3DSSG dataset. Given the high density of nodes and edges in large-scale scenes, abbreviations are employed for edge predicates to ensure visual clarity. The specific mappings are as follows: **a-attached to**, **b-build in**, **c-connected to**, **h-hanging on**, **n-none**, **p-part of**, **s-standing on**, **u-supported by**.

The Fig. 5 presents additional qualitative results on two larger scenes from the 3DSSG dataset, *i.e.*, scene 280d8ebb-6cc6-2788-9153-98959a2da80100 and scene 0cac7580-8d6f-2d13-8c9d-d45247b5244b. For each scene, the reconstructed point cloud is shown on the left, while the ground-truth scene graph, MonoSSG prediction, and our prediction are shown on the right. MonoSSG tends to suffer from object misclassification and relation confusion in complex indoor scenes. For example, in the first scene, several nodes such as window, cabinet, and curtain are incorrectly predicted as wall or table, and some relation edges are also misclassified. In the second scene, MonoSSG similarly confuses categories such as door, cabinet, and furniture, and fails to correctly predict fine-grained relations such as attached to, standing on, and hanging on. In contrast, our method corrects multiple misclassified nodes and recovers more accurate relational edges, producing scene graphs that are more consistent with the ground truth. These results further demonstrate that our method can better distinguish visually similar object categories and capture fine-grained spatial relation patterns in complex 3D scenes.

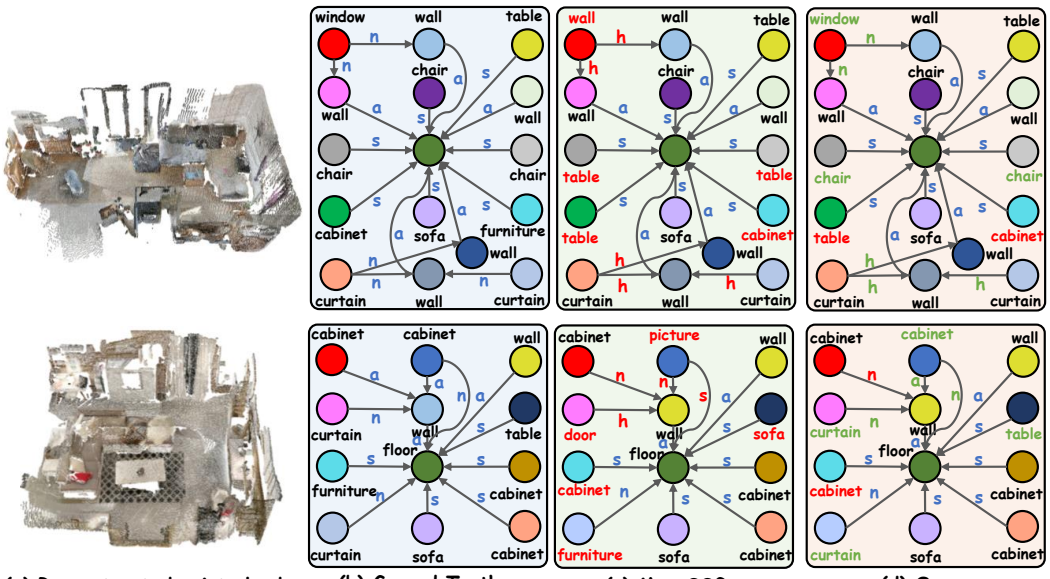

Figure 5: Qualitative results on the 3DSSG [55] dataset. Red edge: misclassified edges from MonoSSG, green edge: edges corrected by our method, red node: misclassified node.

E Discussion

Limitations. Although the proposed hierarchical prototype learning framework achieves promising performance, there remain several aspects worth further exploration. **First,** the learned prototypes are still influenced by the quality of 3D scene observations. In cases of incomplete reconstruction or severe occlusion, node and relation representations may contain limited visual and geometric cues, thereby affecting prototype assignment. **Second,** the proposed hierarchical prototype learning is designed as an auxiliary training strategy and can effectively improve performance even with a lightweight backbone such as ResNet-18. However, this work does not further explore more advanced backbone architectures, which remains an interesting direction for future research. **Finally,** this work has not further explored how to integrate hierarchical prototypes with richer external priors. In the future, fine-grained language priors, object affordance priors, or scene commonsense could be introduced to further enhance the model's understanding of complex scene structures.

Broader Impact. This paper proposes HPSSG, a hierarchical prototype learning framework for RGB-D sequence-based 3D Scene Graph Generation. It is designed to alleviate the inherent inter-class similarity and intra-class diversity of object nodes and relational edges by modeling scene representations in a semantic-to-mode manner. Specifically, semantic-level prototypes provide stable category anchors to enhance global class separability, while mode-level prototypes capture diverse latent patterns within each category to improve fine-grained structural representation. Moreover, category-to-mode prototype alignment maintains hierarchical semantic consistency, and compositional edge prototype learning further exploits subject-object cues to refine relation representations. Our method advances 3D scene understanding and has the potential to benefit various real-world applications, such as robotic navigation, autonomous localization, and embodied scene reasoning. However, similar to many existing 3D vision methods, HPSSG may still suffer from recognition biases when dealing with severely occluded objects, incomplete reconstructions, noisy RGB-D observations, or highly ambiguous spatial relations. In safety-critical autonomous systems, such errors may lead to unreliable relation prediction, incorrect task planning, or potential collision risks. To mitigate these risks, future deployment should incorporate multi-modal safety verification, uncertainty-aware decision making, and human-in-the-loop validation mechanisms to ensure robust and reliable system operation.