SGFormer: Semantic Graph Transformer for Point Cloud-Based 3D Scene Graph Generation

Changsheng Lv[1,2], **Mengshi Qi**[1,2*], **Xia Li**[2], **Zhengyuan Yang**[3], **Huadong Ma**[1,2]
[1]Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia
[2]Beijing University of Posts and Telecommunications
[3]University of Rochester
{lvchangsheng, qms, mhd}@bupt.edu.cn; lixia@bupt.cn; zhengyuan.yang13@gmail.com

Abstract

In this paper, we propose a novel model called SGFormer, *Semantic Graph TransFormer* for point cloud-based 3D scene graph generation. The task aims to parse a point cloud-based scene into a semantic structural graph, with the core challenge of modeling the complex global structure. Existing methods based on graph convolutional networks (GCNs) suffer from the over-smoothing dilemma and can only propagate information from limited neighboring nodes. In contrast, SGFormer uses Transformer layers as the base building block to allow global information passing, with two types of newly-designed layers tailored for the 3D scene graph generation task. Specifically, we introduce the graph embedding layer to best utilize the global information in graph edges while maintaining comparable computation costs. Furthermore, we propose the semantic injection layer to leverage linguistic knowledge from large-scale language model (*i.e.*, ChatGPT), to enhance objects' visual features. We benchmark our SGFormer on the established 3DSSG dataset and achieve a 40.94% absolute improvement in relationship prediction's R@50 and an 88.36% boost on the subset with complex scenes over the state-of-the-art. Our analyses further show SGFormer's superiority in the long-tail and zero-shot scenarios. Our source code is available at https://github.com/Andy20178/SGFormer.

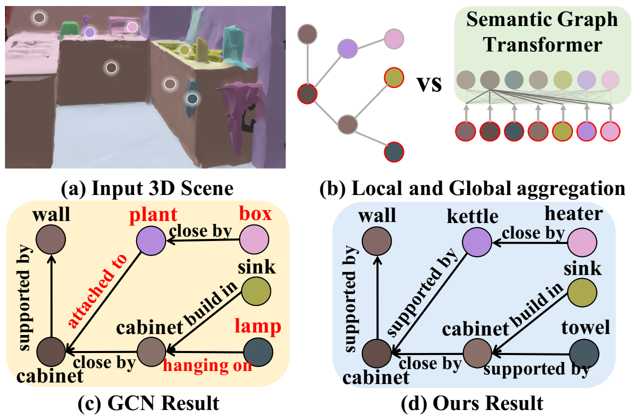

Figure 1: An example of 3D scene graph generation (SGG). (a) an input 3D scene, (b) global aggregation using Transformer on the right *versus* local aggregation using GCN on the left, (c) result using EdgeGCN (Zhang et al. 2021a), and (d) result using our SGFormer. The incorrect results are highlighted in red font, whereas the correct ones are shown in black.

Introduction

Understanding a 3D scene is the essence of human vision, requiring accurate recognition of each object's category and localization, as well as the complex intrinsic structural and semantic relationship. Conventional 3D scene understanding tasks, such as 3D semantic segmentation (Qi et al. 2017; Engelmann et al. 2017; Rethage et al. 2018; Hou, Dai, and Nießner 2019; Vu et al. 2022), object detection and classification (Qi et al. 2017; Zhao et al. 2019; Zheng et al. 2022), focus primarily on the single object localization and recognition but miss higher-order object relationship information, making it challenging to deploy such 3D understanding models in practice. To close this gap, the 3D Scene Graph Generation task (Wald et al. 2020) is recently proposed, as a visually-grounded graph over the detected object instances with edges depicting their pairwise relationships (Yu et al. 2017) (as shown in Figure 1). The significant importance of 3D scene graph is evident as it has already been applied in a wide range of 3D vision tasks, such as VR/AR (Tahara et al. 2020), 3D scene synthesis (Dhamo et al. 2021), and robot navigation (Gomez et al. 2020).

One core challenge for 3D scene graph generation is to accurately predict complex 3D scene structures from sparse and noisy point cloud inputs. Most existing methods (Wald et al. 2020; Zhang et al. 2021a,b; Chen et al. 2022) are built based on Graph Convolution Networks (GCNs) (Kipf and Welling 2016), where nodes represent objects in the scene and edges indicate relationships between objects. Despite the recent success, GCN suffers from its inherent limitation of the over-smoothing dilemma (Li, Han, and Wu 2018). That is, GCN is effective in modeling neighboring nodes with a controlled number of layers but could struggle with learning the global structure and high-order relationships of the entire scene. As shown in Figure 1, the GCN-based method deteriorates in relationship accuracy when the scene

*Corresponding author.

becomes more complicated.

Increasing the GCN layers or node radius could be one attempt to improve the global modeling capability, but often makes the network difficult to converge and results in worse performance, as discussed in previous studies (Ying et al. 2021) and our later analyses. In contrast, we explore a more fundamental change, *i.e.*, replacing the base building block from GCN to Transformer layers (Vaswani et al. 2017), which has shown a strong global context modeling capability and therefore could overcome the limitations of GCNs.

With the above analysis, we propose a Transformer-based model called Semantic Graph TransFormer (SGFormer), to generate a scene graph for a 3D point clouds scene. SGFormer takes both node and edge proposal features obtained from PointNet (Qi et al. 2017) as inputs and stacks Transformer layers to model the higher-order inter-object relationships. The node and edge classifiers then predict each node and edge's categories. However, one technical challenge is the number of edges will grow quadratically to the number of nodes, leading to a quadratically growing input sequence length to the Transformer. To address this challenge, we propose the Graph Embedding Layer that injects the edge information only to the relevant nodes via edge-aware self-attention. In this way, SGFormer preserves the global context with a comparable computation cost.

Furthermore, we propose to exploit external semantic knowledge via the Semantic Injection Layer. Previous GCN-based methods (Zhang et al. 2021b; Chen et al. 2022) have shown the benefit of learning categorical objects and relationships semantic knowledge embeddings. However, these works need to train an extra module to obtain such knowledge. Moreover, the prior knowledge learned from the training set may only work well for common categories in training set distribution but may fail to help with the long-tail object and relationship categories. Alternatively, we design a Semantic Injection Layer to enhance the visual features of object nodes. Specifically, we expand the objects' category labels to detailed descriptions generated from the pre-trained large-scale language model (LLM) (Brown et al. 2020; Touvron et al. 2023), and extract the text features as semantic knowledge embeddings. Note that our approach requires no extra module and significantly improves SGFormer's performance in long-tail and zero-shot scenarios.

Our main contributions can be summarized as follows:

(1) We propose a Semantic Graph Transformer (SGFormer) for 3D scene graph generation, which captures global dependencies between objects and models inter-object relationships with the Graph Embedding Layer. To the best of our knowledge, SGFormer is the first Transformer-based framework for this specific task.

(2) We introduce a Semantic Injection Layer to enhance object features with natural language knowledge and show its effectiveness in the long-tail and zero-shot scenarios.

(3) We benchmark our SGFormer on the established 3DSSG dataset (Wald et al. 2020), with 40.94% absolute improvements over the state-of-the-art GCN-based method. More importantly, we achieved an 88.36% improvement on the subset with complex scenes.

Related Work

Scene Graph Generation. Scene graph was first introduced into image retrieval (Johnson et al. 2015) to capture more semantic information about objects and their inter-relationships. Afterward, the first large-scale dataset, Visual Genome (Krishna et al. 2017), with scene graph annotations on 2D images gave rise to a line of deep learning-based advances (Xu et al. 2017; Li et al. 2017; Herzig et al. 2018; Yang et al. 2018; Zellers et al. 2018; Qi et al. 2019a), and also contributed to the research of other downstream tasks (Qi et al. 2018, 2019a,b, 2020, 2021a,b). Nowadays, to understand the complex 3D indoor structure, 3DSSG dataset (Wald et al. 2020) was first presented to tackle 3D scene graph (Zhang et al. 2021a,b) from point clouds. The EdgeGCN (Zhang et al. 2021a) introduced an edge-oriented GCN to learn a pair of twinning interactions between nodes and edges, and (Zhang et al. 2021b) divide the generation task into two stages, the prior knowledge learning stage and the scene graph generation with a prior knowledge intervention. However, the aforementioned methods fall short in modeling the global-level structure of scenes, and increasing the number of GCN layers easily results in over-smoothing problems. We address the task differently by utilizing the edge-aware self-attention in the Graph Embedding Layer to capture adaptable global representation among nodes.

Knowledge Representation. There has been growing interest in improving data-driven models with external semantic knowledge in natural language processing (Yang et al. 2018; Hinton et al. 2015) and computer vision (Deng et al. 2014; Li, Su, and Zhu 2017; Qi et al. 2019c; Lv et al. 2023), by incorporating semantic cues, (*e.g.,* language priors of object class names) into visual contents (*e.g.,* object proposals) could significantly improve the generation capability (Pennington, Socher, and Manning 2014; Lu et al. 2016; Speer, Chin, and Havasi 2017; Liang et al. 2018). Early method (Zellers et al. 2018) uses statistical co-occurrence as extra knowledge for scene graph generation by introducing a pre-computed bias into the final prediction. In this work, we propose a simple yet effective Semantic Injection Layer to enhance visual features with semantic knowledge getting from LLMs-expanded description by cross-attention mechanism.

Proposed Approach

Overview

Problem Definition. We define a *3D scene graph* as $G = (V, E)$, which describes the category of each object and the corresponding semantic inter-object relationships. In the graph, nodes V refer to the object set, while edges E mean inter-object relationships. Meanwhile, we define the output object class labels as $O = \{o_1, ..., o_N\}, o_i \in \mathcal{C}_{\text{node}}$, where $\mathcal{C}_{\text{node}}$ represents all possible object categories, N is the number of nodes. And the set of inter-object relationships can be defined as $R = \{r_1, ..., r_M\}$, $r_j \in \mathcal{C}_{\text{edge}}$, where $\mathcal{C}_{\text{edge}}$ is the set of all pre-defined relationships classes, M denotes the number of valid edges.

As illustrated in Figure 2, the overall framework of our proposed method follows a typical Transformer architecture

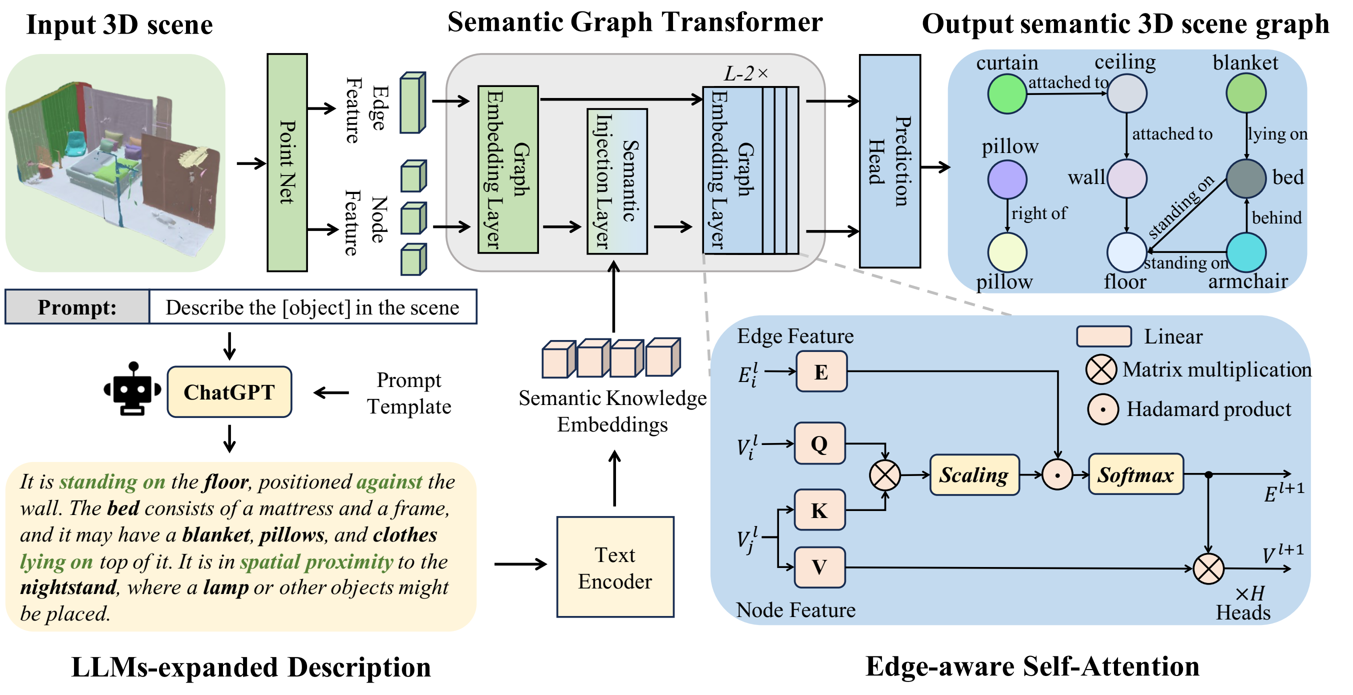
Input 3D scene
Point Net
Edge Feature
Node Feature
Semantic Graph Transformer
Graph Embedding Layer
Semantic Injection Layer
L-2×
Graph Embedding Layer
Prediction Head
Output semantic 3D scene graph
curtain
ceiling
blanket
attached to
attached to
lying on
pillow
wall
bed
right of
standing on
behind
pillow
floor
standing on
armchair
Prompt:
Describe the [object] in the scene
ChatGPT
Prompt Template
Semantic Knowledge Embeddings
Text Encoder
It is standing on the floor, positioned against the wall. The bed consists of a mattress and a frame, and it may have a blanket, pillows, and clothes lying on top of it. It is in spatial proximity to the nightstand, where a lamp or other objects might be placed.
LLMs-expanded Description
Edge Feature
Linear
Matrix multiplication
Hadamard product
E
Q
K
V
Scaling
Softmax
E_i^l
V_i^l
V_j^l
E^{l+1}
V^{l+1}
×H Heads
Node Feature
Edge-aware Self-Attention

Figure 2: Overview pipeline of our proposed SGFormer. We leverage PointNet to initialize the node and edge features in the 3D scene and use LLMs (*i.e.*, ChatGPT) to enrich the object description text of the dataset as semantic knowledge. The SGFormer main consists of two carefully designed components: a Graph Embedding Layer and a Semantic Injection Layer.

but consists of two carefully designed components: *Graph Embedding Layer* **(GEL)** encodes the global-level features of nodes simultaneously with edge-aware self-attention; *Semantic Injection Layer* **(SIL)** extracts the objects' semantic knowledge from the LLMs-expanded descriptions and enhances the node features with knowledge via cross-attention mechanism. For our SGFormer, we empirically set the L layers, *i.e.* a GEL layer and a SIL layer followed by another $L-2$ layers of GEL. We detail the analyses on L and layer setup in ablation studies.

Scene Graph Initialization

Different from 3D instance segmentation (Vu et al. 2022), we study the higher-order problem of object relationship prediction in the scene. Note that we use point cloud data with real instance indexes but without category labels. We follow (Zhang et al. 2021a) adopt the PointNet (Qi et al. 2017) backbone to capture the point-wise feature $\mathbf{X}_P \in \mathbb{R}^{P\times C_{\text{point}}}$ from the input point cloud $\mathbf{P} \in \mathbb{R}^{P\times C_{\text{input}}}$ that forms 3D scene S, where C_{point} and C_{input} denote the channel numbers of point clouds and their extracted point-wise feature, respectively, and P denote the number of sampling points [1].

Node and edge feature generation. Following (Zhang et al. 2021a), for the input scene S, we use the average pooling function (Qi et al. 2017) to aggregate the points in $\mathbf{X}_P$ with the same instance index to obtain the corresponding node visual features $\mathbf{X}_V \in \mathbb{R}^{N\times d_{\text{node}}}$, where N indicates the number of instances in scene S, and d_{node} means node feature dimensions.

Unlike (Wald et al. 2020) uses independent PointNet to extract inter-object relationships, we assume that all objects are connected to each other, and thus we can obtain multi-dimensional edge features based on node features. For each $\mathbf{X}_{E_{(i,j)}} \in \mathbb{R}^{d_{\text{edge}}}$, it denotes the feature for the edge $E_{(i,j)}$ that connects two points from subject V_i toward object V_j, and the features can be initialized as the following formula using the concatenation scheme introduced in (Wang et al. 2019):

$$\mathbf{X}_{E(i,j)} = (\mathbf{X}_{V_i} \parallel (\mathbf{X}_{V_j} - \mathbf{X}_{V_i})), \tag{1}$$

where $\parallel$ denotes the concatenation operation.

Graph Embedding Layer

Following (Dwivedi and Bresson 2021), we feed the input node and edge features into the Graph Embedding Layer. The input node features $\mathbf{X}_{V_i} \in \mathbb{R}^{d_{\text{node}}}$ and edge features $\mathbf{X}_{E_{(i,j)}} \in \mathbb{R}^{d_{\text{edge}}}$ are passed via linear projections to embed into d-dimensional hidden features $\mathbf{V}_i^0$ and $\mathbf{E}_{(i,j)}^0$, respectively.

Multi-Head Edge-aware Self-Attention is proposed in the layer for the message passing in the graph, which is different from the conventional self-attention described in (Vaswani et al. 2017), as shown in Figure 2. For the l-th layer, we use the node feature $\mathbf{V}_i^l$ as query and the neighboring node features $\mathbf{V}_j^l$ $(j = 1, 2, \cdots, N)$ as keys and val-

[1]In our experiments, we set $P = 4096$.

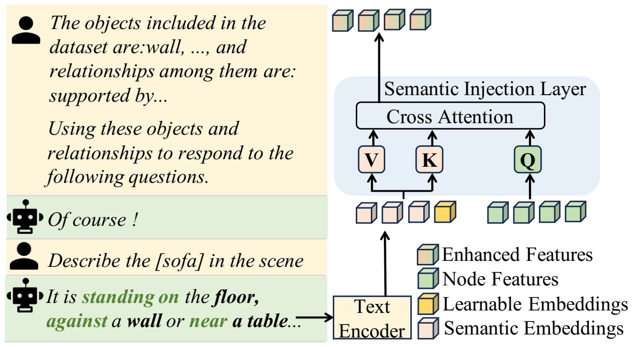
The objects included in the dataset are:wall, ..., and relationships among them are: supported by...
Using these objects and relationships to respond to the following questions.
Of course !
Describe the [sofa] in the scene
It is standing on the floor, against a wall or near a table...
Text Encoder
Semantic Injection Layer
Cross Attention
V
K
Q
Enhanced Features
Node Features
Learnable Embeddings
Semantic Embeddings

Figure 3: Illustration of the prompt templates to LLMs and the proposed Semantic Injection Layer SIL. The enhanced features obtained through SIL will serve as the subsequent layer's node features.

ues. The node features $\mathbf{V}_i^{l+1}$ in the $l+1$ layer are calculated as the concatenation of the self-attention results from H heads. Besides, the updated edge feature $\mathbf{E}_{(i,j)}^{l+1}$ can be calculated by concatenating the edge-aware self-attention maps $\mathbf{M}_{ij}^{l,h} \in \mathbb{R}^{d_h}, 1 \leq h \leq H$, h denotes the number of attention heads and d_h denotes the dimension corresponding to each head. Note that, $\mathbf{M}_{ij}^{l,h}$ in our method is a vector instead of a scalar as in the standard Transformer. The information propagating from node $\mathbf{V}_j$ to $\mathbf{V}_i$ can be formulated as the follows in the layer l:

$$\hat{\mathbf{V}}_i^{l+1} = \mathbf{O}_v^l \cdot \left[\|_{h=1}^{H} \left(\sum_{j}^{N} \mathbf{M}_{ij}^{l,h} \circ \mathbf{W}_V^{l,h} \mathbf{V}_j^{l,h} \right) \right], \quad (2)$$

$$\hat{\mathbf{E}}_{(i,j)}^{l+1} = \mathbf{O}_e^l \cdot \left[\|_{h=1}^{H} \mathbf{M}_{ij}^{l,h} \right], \quad (3)$$

where

$$\mathbf{M}_{ij}^{l,h} = \mathrm{softmax}_j \left(\hat{\mathbf{M}}_{ij}^{l,h} \right), \quad (4)$$

$$\hat{\mathbf{M}}_{ij}^{l,h} = \left(\frac{(\mathbf{W}_Q^{l,h} \mathbf{V}_i^{l,h})^T \cdot \mathbf{W}_K^{l,h} \mathbf{V}_j^{l,h}}{\sqrt{d_h}} \right) \cdot \mathbf{W}_E^{l,h} \mathbf{E}_{i,j}^{l,h}, \quad (5)$$

where $\mathbf{W}_Q^{l,h}, \mathbf{W}_K^{l,h}, \mathbf{W}_V^{l,h}, \mathbf{W}_E^{l,h} \in \mathbb{R}^{d_h \times d_h}, \mathbf{O}_v^l, \mathbf{O}_e^l \in \mathbb{R}^{d \times d}$, $\mathbf{O}_v^l, \mathbf{O}_e^l$ are the weights of linear layers, $\circ$ denotes the Hadamard product, $\cdot$ denotes the matrix multiplication,and $\|$ denotes the concatenation operation.

Following (Dwivedi and Bresson 2021), in order to keep the numerical stability, the outputs after taking exponents of the terms inside softmax will be clamped to a value in $[-5, 5]$. The outputs $\hat{\mathbf{V}}_i^{l+1}$ and $\hat{\mathbf{E}}_{ij}^{l+1}$ are then separately passed into feed-forward network (FFN) preceded and succeeded by residual connections and normalization layers. More detail about FFN please refer to (Vaswani et al. 2017).

Semantic Injection Layer

To fully exploit semantic knowledge within the text modality, we design a new prompt template to obtain a detailed description of the object using the LLMs, *i.e.*, ChatGPT (Brown et al. 2020). After extracting textual features from the descriptions using a text encoder, *i.e.*, CLIP (Radford et al. 2021), we fuse them together with visual features by the Cross-Attention mechanism.

Prompt Template. The output of LLMs provides fine-grained descriptions of each object. To align such descriptions with the content of the given dataset, we design prompt templates that include objects and relationships in the dataset. The prompt template is formulated as *"Within the 3D indoor scene dataset, the objects included in it are: [object], and the relationships among them include [relationship]."*, where [object] and [relationship] represent all object and relationship categories in the dataset. For different objects, we design a prompt as follows: *"Describe the [object] in the scene"*. For example, we input *"Describe the [floor] in the scene"*, obtaining ChatGPT response as:*"The floor in this 3D scene supporting all the other objects in the room. It lies beneath the bed, chair, sofa, table, and other furniture. The floor is in spatial proximity to the wall, door, and window. It is covered by a rug or carpet, and there might be a towel or clothes lying on it."*

As shown in Figure 3, we obtain descriptions of all objects in the dataset using the aforementioned prompts and extract the semantic embeddings $\mathbf{Z}_K \in \mathbb{R}^{K \times d_{\mathrm{emb}}}$ using a frozen pre-trained text encoder, where K is the number of object classes. And for the unknown object categories that don't have corresponding descriptions, we set learnable embeddings $\mathbf{Z}_U \in \mathbb{R}^{U \times d_{\mathrm{emb}}}$ to represent their text features, where $K + U = |\mathcal{C}_{\mathrm{node}}|$ is the category number and d_{emb} denotes the text features dimension. The semantic knowledge embeddings set can be formed as follows:

$$\mathbf{Z} = \mathrm{Concat}(\mathbf{Z}_K, \mathbf{Z}_U). \quad (6)$$

where Concat refers to concatenation in the column dimension, with the $\mathbf{Z} \in \mathbb{R}^{|\mathcal{C}_{\mathrm{node}}| \times d_{\mathrm{emb}}}$ means the semantic knowledge embeddings of all object categories.

Cross-Attention is designed in the layer to aggregate the visual features and the semantic knowledge embeddings followed by two feed-forward networks. Since the node visual features and the semantic knowledge embeddings are in different latent spaces, we transform node features through a layer of a feed-forward network before fusing them. Then for the node i in l-th layer, we use the node feature $\mathbf{V}_i^l$ as query and semantic knowledge embeddings $\mathbf{Z}_i$ as keys and values. We calculate the cross-attention scores $\mathbf{S}_i$ between $\mathbf{V}_i^l$ and each semantic knowledge embedding $\mathbf{Z}_j$ as follows:

$$\mathbf{H}_i^l = \mathrm{Norm}\left(\mathrm{FFN}_H \left(\mathbf{V}_i^l \right) + \mathbf{V}_i^l \right), \quad (7)$$

$$\mathbf{S}_{i,j}^l = \mathrm{softmax}_j \left(\frac{(\mathbf{W}_h \mathbf{H}_i^l)^T \cdot \mathbf{W}_z \mathbf{Z}_j}{\sqrt{d}} \right), \quad (8)$$

where $\mathbf{W}_h \in \mathbb{R}^{d \times d}$, and $\mathbf{W}_z \in \mathbb{R}^{d \times d_{\mathrm{emb}}}$. Therefore the enhanced node features with semantic knowledge injection can be calculated as follows:

$$\hat{\mathbf{U}}_i^l = \sum_j \mathbf{S}_{i,j}^l \cdot \mathbf{W}_u \mathbf{Z}_j^l, \quad (9)$$

$$\mathbf{U}_i^l = \mathrm{Norm}\left(\mathrm{FFN}_U \left(\hat{\mathbf{U}}_i^l \right) + \hat{\mathbf{U}}_i^l \right), \quad (10)$$

where $\mathbf{W}_u \in \mathbb{R}^{d \times d_{\mathrm{emb}}}$.

Training and Inference

As shown in Figure 2, we first initialize the nodes and edges features and extract embeddings from the LLMs-expanded description of each object to get semantic knowledge embeddings. Then Graph Embedding Layer is used to encode and propagate the global node-edge information, while the Semantic Injection Layer is utilized after the first Graph Embedding Layer to inject the semantic knowledge into the node features. The final result is output from the last Graph Embedding Layer, by the prediction head consisting of two MLPs named NodeMLP and EdgeMLP to recognize objects and their structural relationships, respectively.

Specifically, we adopt the focal loss (Lin et al. 2017) for objects and predicates classification due to the data distribution imbalance problem (Zhang et al. 2021b; Wald et al. 2020). We convert object labels into one-hot label and denote prediction logits p as the model's estimated probability for the class where the label $y = 1$. Here we define p_t:

$$p_t = \begin{cases} p, & \text{if } y = 1 \\ 1 - p, & \text{otherwise.} \end{cases} \tag{11}$$

Hence we formulate the object classification focal loss as the following:

$$\mathcal{L}_{\text{focal}}^{\text{obj}} = \alpha \left(1 - p_t\right)^{\gamma} \log\left(p_t\right), \tag{12}$$

where α is the normalized inverse frequency of objects, and γ is a hyper-parameter. Similarly, we can formulate $\mathcal{L}_{\text{focal}}^{\text{edge}}$ as predicate classification focal loss.

Furthermore, to align the injected semantic knowledge embeddings closely with the corresponding object's visual features, we employ the same ground truth in object classification and denote the score S in Eq. (8) as the estimated probability. Upon deriving S_t from Eq. (11), we can define the semantic similarity focal loss for training the Semantic Injection Layer as the following:

$$\mathcal{L}_{\text{SIL}} = \alpha \left(1 - \mathbf{S}_t\right)^{\gamma} log\left(\mathbf{S}_t\right). \tag{13}$$

Therefore, the final loss L_{SG} can be calculated as the sum of the aforementioned three loss functions:

$$\mathcal{L}_{\text{SG}} = \mathcal{L}_{\text{focal}}^{\text{obj}} + \mathcal{L}_{\text{focal}}^{\text{edge}} + \mathcal{L}_{\text{SIL}}. \tag{14}$$

Experiments

Experimental Settings

3DSSG Dataset (Wald et al. 2020). 3DSSG provides annotated 3D semantic scene graphs for 3RScan (Wald et al. 2019). We follow their RIO27 annotation to evaluate 27 class objects and 16 class relationships in our experiments. For the fair comparison, we follow the same experimental settings in (Zhang et al. 2021a) and split the dataset into 1084/113/113 scenes as train/validation/test sets, respectively.

Metrics. We evaluate our model in terms of object, predicate, and relationship classification. We adopt *Recall@K* (Lu et al. 2016) for object classification, computing the *macro-F1 score* and *Recall@K* for predicate classification due to the imbalanced distribution. Besides, we multiply the classification scores of each subject, predicate, and object, and then use *Recall@K* to evaluate the obtained ordered list of relationship classification. Regarding the long-tailed and zero-shot tasks, we use *mean Recall@K(mR@K)* (Tang et al. 2020) for object and predicate classification, and *Zero-Shot Recall@K* (Lu et al. 2016) for never been observed relationships evaluation.

Compared Methods. We compare our approach with the following methods on 3DSSG benchmarks: only using PointNet (Qi et al. 2017), SGPN (Wald et al. 2020), GloRe_{PC} with the point cloud (Ma et al. 2020), GloRe_{SG} with the scene graph (Chen et al. 2019), and Edge-GCN (Zhang et al. 2021a) [2].

Implementation Details

We implement our model based on Pytorch (Paszke et al. 2019) on a single NVIDIA RTX 3090 GPU. Similar to prior works in 3D scene graph generation (Wald et al. 2020; Zhang et al. 2021a), we choose PointNet (Qi et al. 2017) as the backbone. In the Semantic Injection Layer, we use the aforementioned command template to obtain a description of each object, with the category "objects" employing a learnable embedding. For the text encoder, we use CLIP and set the $d_{\text{emb}} = 512$. We set the default layer number L to be 12, consisting of a pair of GEL and SIL layers followed by $L - 2 = 10$ extra GEL layers.

Results

Quantitative Results. We report the quantitative performance of our proposed model compared with other existing methods in Table 1. For *PointNet alone*, we only use PointNet without any reasoning model for evaluation. SPGN (Wald et al. 2020) adopts the edges between nodes as a kind of node to conduct message propagation with GCN. As shown in Table 1, using GCN for scene graph generation could improve the object classification but harm the predicate or relationship classification, which confirms the empirical findings reported in (Wald et al. 2020; Zhang et al. 2021b) about the over-smoothing issue caused by multi-layer GCNs. Instead, our model captures the global scene structure with the Transformer architecture, alleviating the over-smoothing in multi-layer GCN and surpassing the state-of-the-art Edge-GCN (Zhang et al. 2021a). Furthermore, compared with the global-level model GloRe_{SG}, our proposed model obtains better results by utilizing the information passing between inter-object relationships. Attributing to the introduced Semantic Injection Layer, our model achieves the best result in terms of all evaluation metrics.

Qualitative Results. Figures 4 (a) and (b) illustrate qualitative results comparing the state-of-the-art method with our SGFormer, demonstrating significant advancements in both node and edge prediction. In Figure 4 (b), a scenario is presented where "sink" and "object" have only one neighbor, leading to a performance drop for EdgeGCN. In contrast, our SGFormer leverages global information effectively, accurately predicting the labels of these nodes. Furthermore,

[2]In all experiments, the settings of the above-mentioned methods are adopted from the corresponding papers.

Graph Reasoning Approach	Object Class Prediction		Predicate Class Prediction		Relationship Prediction	
	R@5	R@10	F1@3	F1@5	R@50	R@100
+ SGPN (Wald et al. 2020)	89.61	96.98	63.38	77.79	32.45	41.65
+ GloRe$_{PC}$ (Ma et al. 2020)	84.06	95.17	69.23	80.01	31.87	42.21
+ GloRe$_{SG}$ (Chen et al. 2019)	85.27	96.62	72.57	83.42	29.58	38.64
+ EdgeGCN* (Zhang et al. 2021a)	64.46	84.88	33.34	35.90	12.19	19.69
+ EdgeGCN (Zhang et al. 2021a)	90.70	97.58	78.88	90.86	39.91	48.68
+ **Ours w/o SIL**	92.71	97.67	76.64	77.82	50.67	57.50
+ **Ours Full**	**96.41**	**98.48**	**85.12**	**91.58**	**56.25**	**60.67**

Table 1: Comparisons of our model and existing state-of-the-art methods on 3DSSG (Wald et al. 2020). EdgeGCN* denotes aggregates information with all nodes, while EdgeGCN aggregates information only with nodes that have a known relationship. Ours w/o SIL and Ours Full denote our model without the Semantic Injection Layer and our full model, respectively. The best performances are shown in bold.

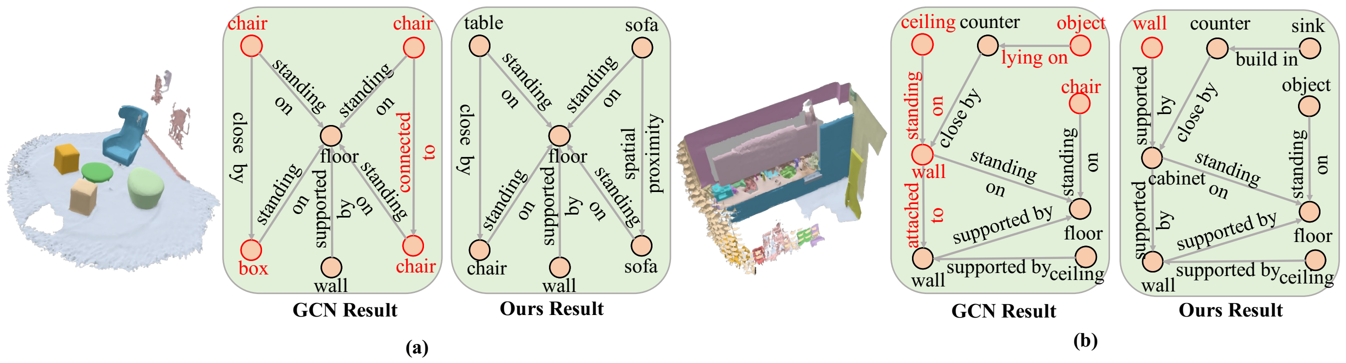

Figure 4: The qualitative results of our model. Given a 3D scene with class-agnostic instance segmentation labels, our SGFormer infers a semantic graph G from the point cloud. For visualization purposes, misclassified object or relationship predictions are indicated in red, while the correct ones are shown in black with the GT value omitted.

Model	Edge	Layer	Node R@1	Edge R@1
Ours full A	×	3	68.34	88.54
Ours full B	✓	3	71.07	91.58
Ours full C	×	6	69.78	89.44
Ours full D	✓	6	70.67	90.89
Ours full E	×	9	71.20	89.85
Ours full F	✓	9	72.50	91.86
Ours full G	×	12	72.53	93.67
Ours full H	✓	12	**75.33**	**96.59**

Table 2: Ablation study on edge feature and layer numbers.

Model	Node		Edge	
	R@1	mR@5	R@1	mR@3
w/o SIL	68.80	84.16	91.07	53.77
SIL$_{\text{Text}}$	65.51	83.67	87.21	53.75
SIL$_{\text{Basic prompt}}$	71.62	88.59	91.70	58.50
SIL$_{\text{GPT w/o template}}$	73.56	90.50	93.47	60.33
SIL$_{\text{GPT w/ template}}$	**75.33**	**92.75**	**96.59**	**65.67**

Table 3: Ablation study for different semantic knowledge.

EdgeGCN exhibits a cascade of errors when reaching an error threshold, while SGFormer maintains robust results in similar situations. These findings emphasize the importance of global information aggregation in SGFormer for enhancing object and relationship recognition in complex scenes. Additionally, when dealing with objects with similar appearances like “cabinet” and “wall”, EdgeGCN struggles with classification based solely on visual cues. However, our approach excels in avoiding such errors by incorporating semantic knowledge, showcasing the effectiveness of our SIL in providing crucial semantic assistance.

Ablation Studies

Graph Embedding Layer (GEL). In Table 2, we design variants of our models to examine the impact of utilizing edge features and the varying layer numbers. Specifically, by comparing the performance of G and H, which have the same depth but differ in node updates, we can observe that incorporating edge features can enhance the model’s predictive capabilities. Additionally, we find that increasing the layer number effectively improves performance, as evidenced by comparing B, D, F, and H, demonstrating the GEL can avoid the over-smoothing problem commonly encountered by GCN when attempting to stack deeper.

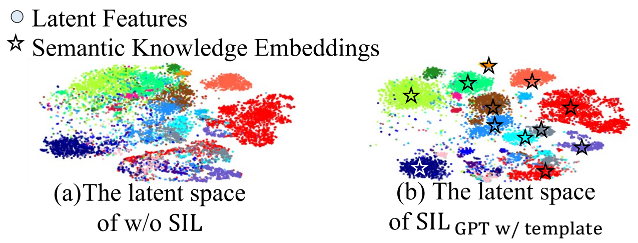

Figure 5: The t-SNE visualization of objects latent space of the w/o SIL(a) and $\mathrm{SIL}_{\mathrm{GPT\ w/\ template}}$(b).

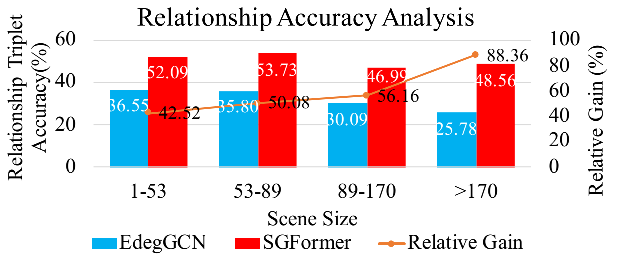

Figure 6: Performance comparison of SGFormer and EdgeGCN w.r.t relationship classification.

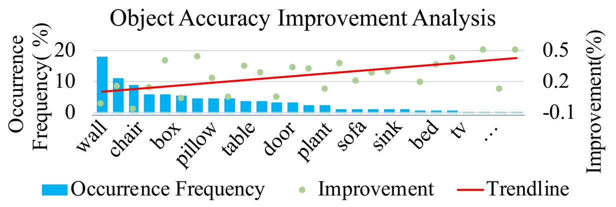

Figure 7: Illustration of objects' per-label accuracy improvement with occurrence frequency in the training set.

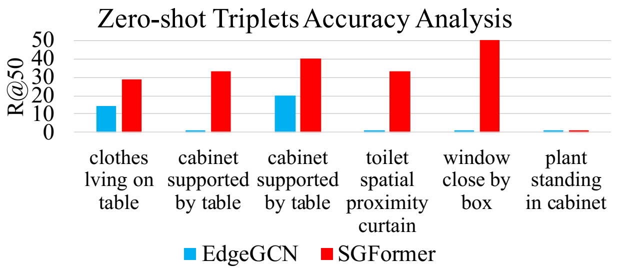

Figure 8: The R@50 results of EdgeGCN and our SGFormer in terms of the zero-shot relationships

Semantic Injection Layer (SIL). We explore various ways to acquire semantic knowledge. To be specific, we directly used object category labels, utilized the basic prompt *"A photo of [object]"*, and obtained descriptions without adding the designed prompt template, denoted as $\mathrm{SIL}_{\mathrm{Text}}$, $\mathrm{SIL}_{\mathrm{Prompt}}$, and $\mathrm{SIL}_{\mathrm{GPT\ w/o\ template}}$, respectively. The results of our experiments are shown in Table 3. Notably, directly extracting word embeddings from category labels resulted in inferior performance compared to SGFormer without SIL and SIL with the basic prompt. Furthermore, we also find that the injected knowledge from the descriptions with adding the proposed prompt templates can help SGFormer achieve the best performance than a simple prompt and descriptions without adding the prompt templates. Besides, Figure 5 demonstrates the latent features and we can see SGFormer with SIL using LLMs-expanded description ($\mathrm{SIL}_{\mathrm{GPT\ w/\ template}}$) achieves a more distinguishable latent space compared with the SGFormer without SIL (w/o SIL), demonstrating that the SIL can enlarge the distance between different object categories, thereby improving object classification accuracy.

How Does SGFormer Help?

Scene size analysis. Defining scene complexity as the summation of the number of nodes and relationships in the whole scene, we compare the performance of our approach with EdgeGCN in varying scene scales. As shown in Figure 6, our SGFormer exhibits a growing relative gain along with the heightened scene complexity.

Long-tail issue. The long-tail phenomenon is very common yet challenging in scene graph generation, where the model usually misleads an uncommon or rare category in the dataset as a common label. As shown in Figure 7, the occurrence frequency of each label in the train set is unbalanced. In the Table 3, $\mathrm{SIL}_{\mathrm{GPT\ w/\ template}}$ achieves the best result, which improves 10.21% and 22.13% than baseline w.r.t Node mR@5 and Edge mR@3, indicating that our proposed SIL can exploit the emergent ability of LLM to improve the predictions performance in marginally sampled object and predicate categories, and successfully alleviate the long-tail challenge. Figure 7 shows the improvement is increasingly greater for very "tail" items. The analysis of predicates can yield the same conclusions.

Zero-shot scenario. In Figure 8. We can observe that our proposed model can improve by a large margin compared with EdgeGCN (Zhang et al. 2021a), such as <cabinet-supported by-table> and <window-close by-box>. These findings consistently demonstrate the efficacy and superiority of the proposed GEL and SIL in our SGFormer. However, both methods perform poorly in <plant-standing in-cabinet> due to this relationship being uncommon in real life, and more training data can resolve this case.

Conclusion

In this paper, we proposed a novel Semantic Graph Transformer model (*i.e.*, SGFormer) for 3D scene graph generation, including Graph Embedding Layer to capture the global-level structure and Semantic Injection Layer to enhance the objects' visual features with LLMs' powerful semantic knowledge. Extensive experiments on the 3DSSG benchmark demonstrate the effectiveness and state-of-the-art performance of our proposed model. Notably, our SGFormer performs exceptionally well on complicated scenes, and under challenging long-tail and zero-shot scenarios.

Acknowledgements

This work was partly supported by the Funds for Creative Research Groups of China under Grant 61921003, the National Natural Science Foundation of China under Grant 62202063, the Young Elite Scientists Sponsorship Program by China Association for Science and Technology (CAST) under Grant 2021QNRC001, the 111 Project under Grant B18008, the Open Project Program of State Key Laboratory of Virtual Reality Technology and Systems, Beihang University (No.VRLAB2022C01).

References

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33: 1877–1901.

Chen, L.; Lu, J.; Cai, Y.; Wang, C.; and He, G. 2022. Exploring Contextual Relationships in 3D Cloud Points by Semantic Knowledge Mining. In *Computer Graphics Forum*, volume 41, 75–86. Wiley Online Library.

Chen, Y.; Rohrbach, M.; Yan, Z.; Shuicheng, Y.; Feng, J.; and Kalantidis, Y. 2019. Graph-based global reasoning networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 433–442.

Deng, J.; Ding, N.; Jia, Y.; Frome, A.; Murphy, K.; Bengio, S.; Li, Y.; Neven, H.; and Adam, H. 2014. Large-scale object classification using label relation graphs. In *Proceedings of the European Conference on Computer Vision*, 48–64.

Dhamo, H.; Manhardt, F.; Navab, N.; and Tombari, F. 2021. Graph-to-3d: End-to-end generation and manipulation of 3d scenes using scene graphs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 16352–16361.

Dwivedi, V. P.; and Bresson, X. 2021. A Generalization of Transformer Networks to Graphs. *AAAI Workshop on Deep Learning on Graphs: Methods and Applications*.

Engelmann, F.; Kontogianni, T.; Hermans, A.; and Leibe, B. 2017. Exploring spatial context for 3D semantic segmentation of point clouds. In *Proceedings of the IEEE international conference on computer vision workshops*, 716–724.

Gomez, C.; Fehr, M.; Millane, A.; Hernandez, A. C.; Nieto, J.; Barber, R.; and Siegwart, R. 2020. Hybrid topological and 3d dense mapping through autonomous exploration for large indoor environments. In *2020 IEEE International Conference on Robotics and Automation*, 9673–9679. IEEE.

Herzig, R.; Raboh, M.; Chechik, G.; Berant, J.; and Globerson, A. 2018. Mapping images to scene graphs with permutation-invariant structured prediction. *Advances in Neural Information Processing Systems*, 31.

Hinton, G.; Vinyals, O.; Dean, J.; et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Hou, J.; Dai, A.; and Nießner, M. 2019. 3d-sis: 3d semantic instance segmentation of rgb-d scans. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4421–4430.

Johnson, J.; Krishna, R.; Stark, M.; Li, L.-J.; Shamma, D.; Bernstein, M.; and Fei-Fei, L. 2015. Image retrieval using scene graphs. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 3668–3678.

Kipf, T. N.; and Welling, M. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Krishna, R.; Zhu, Y.; Groth, O.; Johnson, J.; Hata, K.; Kravitz, J.; Chen, S.; Kalantidis, Y.; Li, L.-J.; Shamma, D. A.; et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1): 32–73.

Li, G.; Su, H.; and Zhu, W. 2017. Incorporating external knowledge to answer open-domain visual questions with dynamic memory networks. *arXiv preprint arXiv:1712.00733*.

Li, Q.; Han, Z.; and Wu, X.-M. 2018. Deeper insights into graph convolutional networks for semi-supervised learning. In *Thirty-Second AAAI conference on Artificial Intelligence*.

Li, Y.; Ouyang, W.; Zhou, B.; Wang, K.; and Wang, X. 2017. Scene graph generation from objects, phrases and region captions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1261–1270.

Liang, K.; Guo, Y.; Chang, H.; and Chen, X. 2018. Visual relationship detection with deep structural ranking. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Lin, T.-Y.; Goyal, P.; Girshick, R.; He, K.; and Dollár, P. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2980–2988.

Lu, C.; Krishna, R.; Bernstein, M.; and Fei-Fei, L. 2016. Visual relationship detection with language priors. In *Proceedings of the European Conference on Computer Vision*, 852–869.

Lv, C.; Zhang, S.; Tian, Y.; Qi, M.; and Ma, H. 2023. Disentangled Counterfactual Learning for Physical Audiovisual Commonsense Reasoning. *arXiv preprint arXiv:2310.19559*.

Ma, Y.; Guo, Y.; Liu, H.; Lei, Y.; and Wen, G. 2020. Global context reasoning for semantic segmentation of 3D point clouds. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2931–2940.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing*, 1532–1543.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 652–660.

Qi, M.; Li, W.; Yang, Z.; Wang, Y.; and Luo, J. 2019a. Attentive relational networks for mapping images to scene graphs.

In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3957–3966.

Qi, M.; Qin, J.; Huang, D.; Shen, Z.; Yang, Y.; and Luo, J. 2021a. Latent memory-augmented graph transformer for visual storytelling. In *Proceedings of the 29th ACM International Conference on Multimedia*, 4892–4901.

Qi, M.; Qin, J.; Li, A.; Wang, Y.; Luo, J.; and Van Gool, L. 2018. stagnet: An attentive semantic rnn for group activity recognition. In *Proceedings of the European Conference on Computer Vision*, 101–117.

Qi, M.; Qin, J.; Yang, Y.; Wang, Y.; and Luo, J. 2021b. Semantics-aware spatial-temporal binaries for cross-modal video retrieval. *IEEE Transactions on Image Processing*, 30: 2989–3004.

Qi, M.; Wang, Y.; Li, A.; and Luo, J. 2019b. Sports video captioning via attentive motion representation and group relationship modeling. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(8): 2617–2633.

Qi, M.; Wang, Y.; Li, A.; and Luo, J. 2020. STC-GAN: Spatio-temporally coupled generative adversarial networks for predictive scene parsing. *IEEE Transactions on Image Processing*, 29: 5420–5430.

Qi, M.; Wang, Y.; Qin, J.; and Li, A. 2019c. KE-GAN: Knowledge embedded generative adversarial networks for semi-supervised scene parsing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5237–5246.

Radford, A.; Kim, J. W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, 8748–8763.

Rethage, D.; Wald, J.; Sturm, J.; Navab, N.; and Tombari, F. 2018. Fully-convolutional point networks for large-scale point clouds. In *Proceedings of the European Conference on Computer Vision*, 596–611.

Speer, R.; Chin, J.; and Havasi, C. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on Artificial Intelligence*.

Tahara, T.; Seno, T.; Narita, G.; and Ishikawa, T. 2020. Retargetable AR: Context-aware augmented reality in indoor scenes based on 3D scene graph. In *2020 IEEE International Symposium on Mixed and Augmented Reality Adjunct*, 249–255. IEEE.

Tang, K.; Niu, Y.; Huang, J.; Shi, J.; and Zhang, H. 2020. Unbiased scene graph generation from biased training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 3716–3725.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

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

Vu, T.; Kim, K.; Luu, T. M.; Nguyen, T.; and Yoo, C. D. 2022. Softgroup for 3d instance segmentation on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2708–2717.

Wald, J.; Avetisyan, A.; Navab, N.; Tombari, F.; and Nießner, M. 2019. RIO: 3D object instance re-localization in changing indoor environments. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7658–7667.

Wald, J.; Dhamo, H.; Navab, N.; and Tombari, F. 2020. Learning 3d semantic scene graphs from 3d indoor reconstructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3961–3970.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics*, 38(5): 1–12.

Xu, D.; Zhu, Y.; Choy, C. B.; and Fei-Fei, L. 2017. Scene graph generation by iterative message passing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 5410–5419.

Yang, J.; Lu, J.; Lee, S.; Batra, D.; and Parikh, D. 2018. Graph r-cnn for scene graph generation. In *Proceedings of the European Conference on Computer Vision*, 670–685.

Ying, C.; Cai, T.; Luo, S.; Zheng, S.; Ke, G.; He, D.; Shen, Y.; and Liu, T.-Y. 2021. Do transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34: 28877–28888.

Yu, R.; Li, A.; Morariu, V. I.; and Davis, L. S. 2017. Visual relationship detection with internal and external linguistic knowledge distillation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 1974–1982.

Zellers, R.; Yatskar, M.; Thomson, S.; and Choi, Y. 2018. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5831–5840.

Zhang, C.; Yu, J.; Song, Y.; and Cai, W. 2021a. Exploiting edge-oriented reasoning for 3d point-based scene graph analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 9705–9715.

Zhang, S.; Hao, A.; Qin, H.; et al. 2021b. Knowledge-inspired 3D Scene Graph Prediction in Point Cloud. *Advances in Neural Information Processing Systems*, 34: 18620–18632.

Zhao, Y.; Birdal, T.; Deng, H.; and Tombari, F. 2019. 3D point capsule networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 1009–1018.

Zheng, Y.; Duan, Y.; Lu, J.; Zhou, J.; and Tian, Q. 2022. HyperDet3D: Learning a Scene-conditioned 3D Object Detector. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5585–5594.