Recognition of Indoor Scenes Using 3-D Scene Graphs

Han Yue, Ville Lehtola, Hangbin Wu, George Vosselman, Jincheng Li, and Chun Liu

***Abstract*— Scene recognition is a fundamental task in 3-D scene understanding. It answers the question, "What is this place?" In an indoor environment, the answer can be an office, kitchen, lobby, and so on. As the number of point clouds increases, using embedded point information in scene recognition becomes computationally heavy to process. To achieve computational efficiency and accurate classification, our idea is to use an indoor scene graph that represents the 3-D spatial structures via object instances. The proposed method comprises two parts, namely: 1) construction of indoor scene graphs leveraging object instances and their spatial relationships and 2) classification of these graphs using a deep learning network. Specifically, each indoor scene is represented by a graph, where each node represents either a structural element (like a ceiling, a wall, or a floor) or a piece of furniture (like a chair or a table), and each edge encodes the spatial relationship between these elements. Then, these graphs are used as input for our proposed graph classification network to learn different scene representations. The public indoor dataset, ScanNet v2, with 625.53 million points, is selected to test our method. Experiments yield good results with up to 88.00% accuracy and 82.30% F1 score in the fixed validation dataset and 90.46% accuracy and 81.45% F1 score in the ten-fold cross-validation method; moreover, if some indoor objects cannot be successfully identified, the scene classification accuracy depends sublinearly on the rate of missing objects in the scene.**

***Index Terms*— Graph classification, indoor, point clouds, scene graphs, scene recognition.**

I. Introduction

With the rapid development and popularization of 3-D sensors, 3-D scene understanding has become a hot research topic in recent years. Scene recognition as a perception task of 3-D scene understanding is related to applications in robotics [1], positioning, and navigation [2]. It answers the question, "What kind of place is this?" For the indoor environment, the answer equals the type of room, such as an office, kitchen, or lobby.

Surprisingly, little attention has been given to performing indoor scene recognition based on 3-D data such as point clouds, although indoor environments are inherently 3-D [3]. Instead, 2-D image-based methods for indoor scene recognition have been investigated for many years [4], [5]. Point clouds of an indoor scene are not input into the network directly because the number of points is too large, making training computationally intractable. Consequently, the first study based on 3-D data by Huang et al. [6] explored two different options using point clouds as input data: blocks and voxels. This trade-off comes with disadvantages. First, 1) splitting the scene into some blocks of points will destroy the spatial relationships between objects that may be essential for scene recognition, resulting in poor accuracy. Second, although using voxel grids derived from point clouds as input can preserve the spatial structures of objects, 2) a lot of memory is required. The reason is that grids need to be dense to get a good result. Last, 3) the above methods only learn one decoration style (such as color, or shape) and have limitations when they are used in another environment with a different style but the same function.

To address the above disadvantages 1)–3), we argue that scene graphs constructed from objects instead of point blocks or voxel grids may offer a better approach for large-scale indoor 3-D scene recognition. It means our method is built on top of computationally light object instances. These objects can be derived from indoor point clouds by instance segmentation or object detection methods. Then, a scene graph is an abstract representation that organizes the entities of a scene in a graph, where objects are nodes and their relationships are modeled as edges. Such a representation is frequently used in the image domain for higher level tasks such as image retrieval [7] and image generation [8]. Recently, scene graphs have started to emerge in 3-D. For example, scene graphs of walls and obstacles have been used for navigation studies [9]. Armeni et al. [10] showed that it was plausible to construct graphs for the whole buildings, including 3-D spatial relations between rooms, major objects, camera views, and the relations between these entities. These graphs can be further incorporated with dynamic elements such as moving humans [11] and semantic relationships such as descriptors front, behind, and standing on [12], [13]. Hence, 3-D scene graphs are becoming more common and more complex, which also increases their usability.

Manuscript received 5 June 2023; revised 20 November 2023 and 19 February 2024; accepted 27 March 2024. Date of publication 11 April 2024; date of current version 25 April 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 42271429 and Grant 42130106, in part by the Fundamental Research Funds for the Central Universities under Grant 22120240133, and in part by the Visiting Fellowship from the China Scholarship Council under Grant 202106260143. *(Corresponding author: Hangbin Wu.)*

Han Yue, Hangbin Wu, and Chun Liu are with the College of Surveying and Geo-Informatics, Tongji University, Shanghai 200092, China (e-mail: yuehan@tongji.edu.cn; hb@tongji.edu.cn; liuchun@tongji.edu.cn).

Ville Lehtola and George Vosselman are with the Department of Earth Observation Science, ITC Faculty, University of Twente, 7522 NB Enschede, The Netherlands (e-mail: v.v.lehtola@utwente.nl; george.vosselman@utwente.nl).

Jincheng Li is with the Key Laboratory of 3-D Information Acquisition and Application, MOE, Capital Normal University, Beijing 100048, China (e-mail: winston1566@163.com).

Digital Object Identifier 10.1109/TGRS.2024.3387556

In this article, we set out to study indoor scene recognition based on 3-D scene graphs. Our inspiration comes from biology, namely, classifying protein functions given their structures [14]. Similarly, we classify the types of indoor scenes given their structures. To be specific, each indoor scene is represented by a 3-D scene graph, which captures the semantic information and spatial adjacency relation between elements that the scene contains. Then, these indoor scene graphs are input into our proposed graph classification network to classify scene types. The main contributions are summarized as follows:

1) We present to recognize indoor scenes based on constructed 3-D scene graphs, addressing the above three disadvantages. In this way, we (a) keep the spatial structure of scenes complete and (b) drastically reduce memory usage by storing objects in a graph as abstractions. Specifically, our lightweight representation allows the use of a global receptive field to model long-range dependencies; moreover, this makes (c) the scene graph invariant to decoration changes, and the scene type is only determined with indoor 3-D scene graphs.
2) We introduce the implementation of an indoor scene graph classification network, which contains two modules: a spatial relations-based local feature aggregation module (SR-LFAM) and an object attention-based global feature aggregation module (OA-GFAM). In the OA-GFAM branch, we design quantity encoding and spatial position encoding to help capture long-range dependencies to enhance the global graph representation.
3) We build an indoor 3-D scene graph dataset from ScanNet v2 point cloud dataset to support the study of indoor scene recognition. A comprehensive performance analysis of these graphs demonstrated that the proposed architecture achieves accurate and efficient indoor scene classification.

The rest of the article is organized as follows. In Section II, we review the related work about 2-D/3-D indoor scene recognition and graph classification networks. In Section III, our proposed method is introduced in detail. The experimental results are reported in Section IV. The article is concluded in Section V.

II. Related Works

A. Indoor Scene Recognition

Research on vision-based scene recognition has been investigated for many years and a lot of successful approaches have been proposed [15], [16], [17], [18], [68]. Several scene-centered datasets are available, such as MIT Indoor67 [5], SUN397 [20], and Places365 [21]. Early works used statistics-based and handcrafted features for scene representation [22], [23], [24], [25], [26], [27]. Oliva and Torralba [22] proposed a low-dimensional global feature descriptor to model holistic spatial scene properties. Valiaya et al. [24] and Serrano et al. [25] combined some low-level visual features, such as color and edge, which can be trained by a support vector machine (SVM) classifier for scene recognition. Lazebnik et al. [19] computed histograms of local features found inside each subregion and aggregated them to get a global scene representation. Xie et al. [26] introduced the orientational pyramid matching model and concluded that the 3-D orientation of objects is a crucial factor in discriminating indoor scenes; nevertheless, these methods are based on low-level features, which are inflexible and hard to design for different images. These shortcomings limit the method's further applications.

The emergence of neural networks brought new opportunities to indoor scene recognition. Sun et al. [17] presented a comprehensive representation of scene images that included object semantics, global appearance, and contextual appearance. These features were extracted from different deep-learning models and finally were fed to the SVM classifier. Song et al. [18] proposed two discriminative image representations for scene recognition, namely, co-occurring frequency of object-to-object relation and sequential representation of object-to-object relation. Yuan et al. [67] presented an adaptive cross-modal GCN to model indoor scenes, where the most distinctive features were selected for visual scene representation. Miao et al. [5] proposed an object-to-scene method, which extracted object features and learned object relations to recognize indoor scenes by training an end-to-end convolutional neural network; however, it is also an image-based scene recognition method, which cannot encode the spatial relationship of objects.

Deep learning methods based on 3-D point clouds mainly focus on the understanding of the semantics and instances of indoor objects rather than on the scene level. The pioneering work on point cloud deep learning is PointNet [28]. After that, many networks were proposed to capture local features and global features in different ways, such as DGCNN [29], KPConv [30], RandLA-Net [31], PointNeXt [32], and Swin3D [33], which obtained good performance in semantic segmentation. At the same time, instance segmentation methods have also received widespread attention [34], [35], [36], [37], [38], [39]. These methods segment individual object instances from indoor scene point clouds and predict their semantics. Various clustering algorithms (such as Mean-Shift) are usually used in these networks. For example, ASIS [34] and JSNet [35] proposed joint instance and semantic segmentation of indoor point clouds with a mean-shift clustering algorithm for postprocessing. After that, PointGroup [36], SoftGroup [37], SPFormer [38], and Mask3D [39] are presented to enhance the performance of instance segmentation. Scene recognition using 3-D point cloud data, however, is yet in its infancy.

Conducting indoor scene recognition in 3-D was started by Huang et al. [6], who split or voxelized the original point clouds in the data preprocessing stage. Specifically, two different options were explored for the encoder: the point-based method and the voxel-based method. The point-based method splits the original point clouds into blocks as input, destroying the completeness and spatial relationship of the objects, while the voxel-based network works with sparse voxel grids derived from the input points with a lot of memory requirements. In addition, the above networks only learn one decoration style (such as color or shape) and have limitations when used

in another environment with a different style but the same function.

B. Graph Classification in Deep Learning

Graph neural networks (GNNs) are used to process unstructured data, such as social networks, chemical molecules, point clouds, and so on. Recently, GNNs have drawn considerable attention, including node classification [41], [42], link prediction [43], [44], and graph classification [45], [46], [47]. As one of the applications in GNNs, the goal of graph classification is to learn a graph-level representation and then predict its label by aggregating node features and graph structure information. The learning process mainly includes two key steps: graph convolution and graph pooling.

Inspired by the great success achieved by the convolutional neural network on images, a graph convolution network (GCN) is used to generalize convolution for graph-based data. It can be divided into two categories: spectral [48], [49], [50] and spatial [46], [51] approaches. Spectral-based methods typically define the parameterized filters according to graph spectral theory. Bruna et al. [48] first proposed to define graph convolution in the Fourier domain based on the Laplacian graph, which had the disadvantage of high computational cost. Following this, some methods have been proposed to address this challenge. Defferrard et al. [49] presented ChebNet and improved its efficiency by designing fast localized convolutional filters on a graph through Chebyshev expansion. GCN is further simplified by Kipf and Welling [41]. This network can encode both the local graph structure and features of nodes. In contrast, spatial approaches define graph convolution by directly aggregating the node's neighborhood information based on edge connections. In this way, each node embedding can be updated from its neighbors. For example, Hamilton et al. [51] proposed a framework called "GraphSage" to generate embeddings by sampling and aggregating features from a node's local neighborhood. GAT [51] specified different weights to different nodes in a neighborhood and aggregated them to generate the central node embeddings.

Graph pooling is a critical step in the graph classification task to scale down the size of inputs and then obtain a graph-level representation from learned node embeddings. Previous works [52] and [53] used global pooling operations (such as sum/average/max pooling) over the node representations, which could lose a lot of structure information. In recent years, the hierarchical graph pooling approach has emerged.

Ying et al. [47] proposed hierarchical graph representation learning with differentiable pooling (DiffPool). This method assigns nodes to a set of clusters (namely, subgraphs), and each subgraph can be aggregated as a super node. These nodes form a new coarsened graph with a full connection, which is fed to the next GCN layer. After that, MxPool [14] enriched the DiffPool method, which can adapt to different sizes of graph data and select optimal coarsening ratio and feature dimension based on the attention mechanism. SAGPool [54], gPool [55], and AttPool [56] were all developed based on the Top-K method. To be specific, an attention score is calculated for each node in the graph, which is then sorted and the top k nodes are selected to form a new subgraph, which will be entered into the subsequent layer for the message-passing process.

Recently, applying transformer model in graphs has gained popularity in the field of graph representation learning [50], [63], [64], [65]. These networks attempt to incorporate various encodings (such as positional encoding and structural encoding) into self-attention mechanism to capture the local and global information of graphs. Dwivedi and Bresson [63] introduced transformer model into graph representation and used Laplacian eigenvector as position encoder to learn graph structures and extended this architecture to edge feature representation. Chen et al. [64] proposed a new self-attention mechanism by extracting a subgraph representation rooted at each node before computing the attention. Our method is also based on the self-attention module. We designed two simple but effective encoding designs to help learn the indoor scene graph representation well.

III. Methodology

A. Overview of the Method

The proposed method consists of two parts (shown in Fig. 1): indoor scene graph construction (ISGC) and indoor scene graph classification. The input is 3-D object instances with coordinates and semantics like chairs, tables, and doors. These objects can be derived from indoor point clouds by manual labeling, instance segmentation, or object detection. In this article, the ScanNet v2 dataset is adopted for scene recognition, where all semantics and instances are already provided. The output of the method is the predicted indoor scene type, such as a conference room, an office, a hallway, etc.

Indoor scene type is related to the objects it contains and the mutual spatial relationship between these objects; therefore, we first present ISGC method to create a graph from object instances contained in an indoor scene. In the construction phase, a graph representation is formulated by our method, where connections are established depending on the object-pair semantics and the geometrical distance between the pair. After this, we propose an indoor scene classification network, which learns different graph representations through SR-LFAM and the OA-GFAM.

1) Problem Setting: We expect each scene to consist of a single room. If a point cloud covers multiple rooms, a space partitioning algorithm needs to be used to split the dataset into one point cloud per room [60] and [61], given the point clouds of such an indoor scene, where all 3-D objects are with semantics and (x, y, z) coordinates. Then, create a set of indoor scene graphs and classify them to predict scene types.

Let an indoor graph be represented as follows:

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

where V is the set of nodes representing indoor objects (like walls, chairs, etc.) and E is the set of edges, representing the spatial relationships between nodes in a graph. $A \in R^{n\times n}$ denotes the adjacency matrix of G and $X \in R^{n\times d}$ represents the node feature matrix, where d is the dimension of node

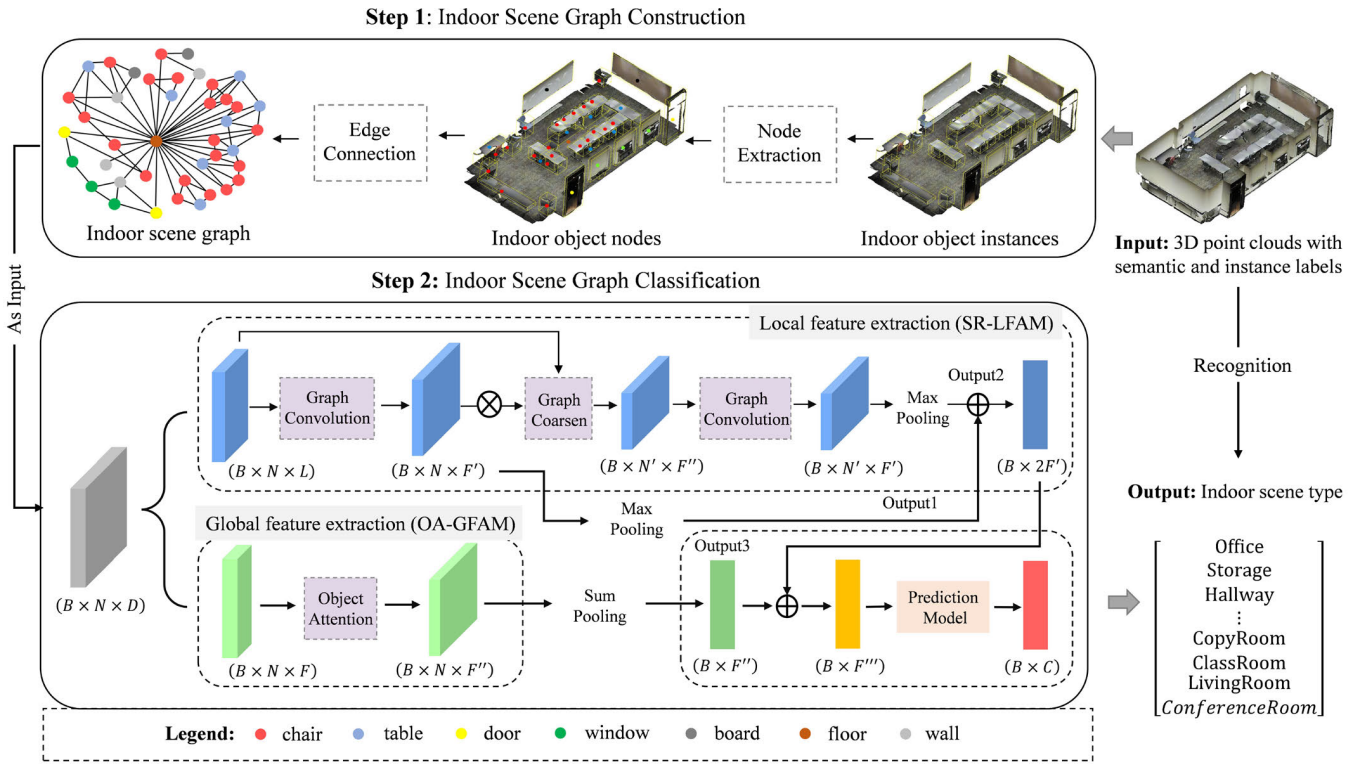


Fig. 1. Illustration of the proposed method.

features. Specifically, $A_{i,j} = 1$ represents there is an edge between nodes i and j, otherwise, $A_{i,j} = 0$. We suppose each node belongs to one of the C indoor object classes. Then, let an indoor graph dataset $G_{\text{indoor}} = \{G_1, G_2, \ldots, G_m\}$, where each graph G_i represents an indoor scene. The aim is to predict a corresponding semantic label for each graph; therefore, these graphs G_{indoor} are fed into a graph classification network and the scene types are predicted.

As shown in Fig. 1, given a set of graphs $G = \{G_1, G_2, \ldots, G_m\}$ as input. The data shape is $(B \times N \times D)$, where B and N represent the number of graphs and object nodes in a graph, respectively. D is feature vector at node v_i. Then we feed different graph features $(B \times N \times L), (B \times N \times F)$ into subbranches to learn local and global graph representations. L and F represent the different node feature dimension. Finally, contact local and global graph embeddings and input it into multilayers perceptron (MLP) to obtain predicted graph types $(B \times C)$, where C is the number of scene types. In this way, the mapping from indoor point clouds to indoor scene type is realized. The method details are described in Sections III-B and III-C.

B. Indoor Scene Graph Construction

In this section, the ISGC method named ISGC is proposed to represent an indoor scene by a graph from labeled 3-D point clouds (all points with semantic and instance labels). There are two key parts in this algorithm: graph node extraction and edge connection.

1) Graph Node Extraction: The proposed method is based on given object instances derived from indoor scene point clouds. This means each point has its corresponding semantic and instance label. Bounding boxes can, therefore, be detected directly based on these indoor objects. The center of each bounding box is considered an indoor object, that is, a node in the graph. To be specific, each object instance I_i is represented by node v_i, with its semantic class l_i and the center coordinate (x_i, y_i, z_i) of the object 3-D bounding box, as the node features.

The process of extracting object nodes from labeled indoor point clouds is shown in Fig. 2. The interior object nodes contained in a scene can be divided into structural elements (ceiling, floor, wall, etc.) and furniture elements (chair, table, sofa, bookcase, etc.).

2) Graph Edge Connection: Edges E need to be created to establish links between object nodes in set V (obtained from Section III-B1). Each edge represents the spatial relationship between two objects it connects. A complete indoor scene graph is shown in Fig. 3. For visual clarity and verbal convenience, we explain the formation of edges in this graph by using two subgraphs, namely the structural element subgraph and the furniture element subgraph.

The structural element subgraph contains walls, ceilings, and floors as nodes [see Fig. 3(a)]. Links are built based on prior spatial adjacency knowledge. For example, the ceiling and the floor nodes are usually connected to the wall nodes. Different walls are usually connected if their 3-D bounding boxes are intersected.

The furniture element subgraph contains the pieces of furniture as nodes [see Fig. 3(b)]. We believe that indoor objects are related to objects within a certain distance; therefore, a set of nearest neighbors are found for each node with a physical distance threshold and then connected to them. Here, detect-

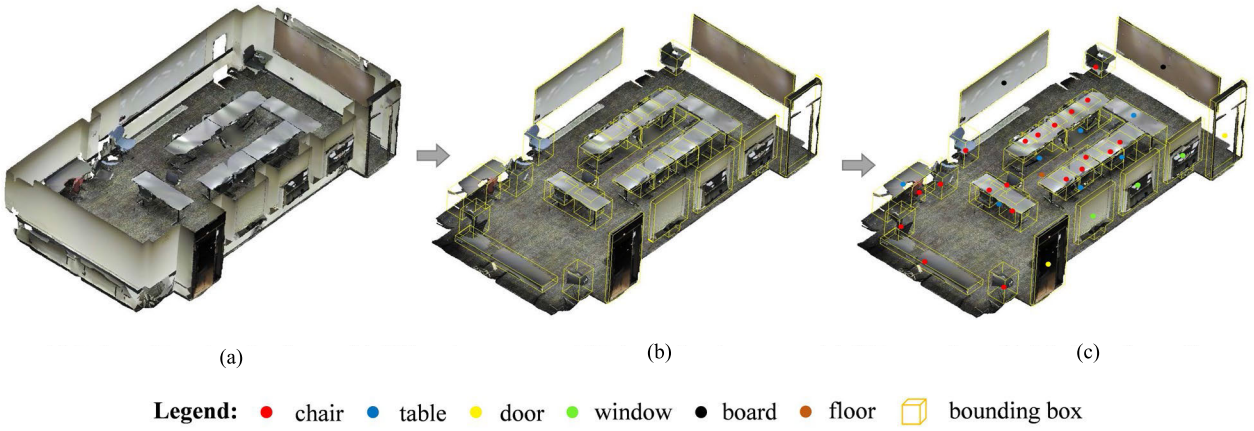


Fig. 2. Graph node extraction form labeled 3-D point clouds [Walls are removed for clarity in (b) and (c)]. (a) Indoor 3-D point clouds, (b) Object instances and 3-D bounding boxes, and (c) Object nodes with labels and coordinates.

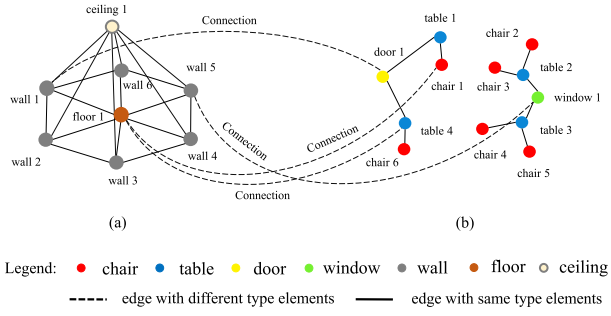


Fig. 3. Indoor scene graph. (For visual clarity purposes, show the graph using two subgraphs.) (a) Structural elements (sub graph1) and (b) Furniture elements (sub graph2).

ing the central node's neighbors by calculating whether the 3-D bounding box buffers intersect instead of the k nearest neighbors (K-NN) algorithm. This is because the size of indoor scenes varies greatly; therefore, it is difficult to find a fixed k to describe the neighbors of objects.

We merge the structural element subgraph and furniture element subgraph to form the entire indoor scene graph by introducing a link wherever the floor or wall is close to a furniture element.

In this way, our graph can represent a complete indoor scene structure by storing objects in a graph as abstractions instead of directly processing point clouds. The node features only contain object semantic l_i and coordinate (x_i, y_i, z_i). Its advantage is to make the scene graph invariant to decoration changes like color or shape.

C. Indoor Scene Graph Classification Network

In this article, indoor 3-D scene recognition is achieved by an indoor scene graph classification task. Graph classification aims to obtain a graph-level representation from learned node embeddings. Two branches are, therefore, proposed to generate different graph representations, namely the SR-LFAM and the OA-GFAM.

1) Spatial Relations-Based Local Feature Aggregation Module: In this module, a graph representation is obtained by graph convolution and hierarchical graph pooling operation to learn spatial relations between objects (see Fig. 4). To be specific, each node embedding can be updated from its neighbor nodes based on edge connections (graph convolution), and then we scale down the size of the graph by coarsen ratios (graph pooling) until there is one graph node left.

a) Graph convolution: For each node, the features of its neighbors are aggregated through GCN [41] to update the node embeddings. The kth layer in GCN takes the graph adjacency matrix A and the hidden representation matrix H_k as input, then the next layer's output will be generated as follows (a single convolutional layer):

$$H_{k+1} = \sigma\left(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H_k W^k\right) \tag{2}$$

where $\sigma(\cdot)$ is the nonlinear activation function and H_{k+1} are the node embeddings computed after $k+1$ layers, $\tilde{A} = A + I$ is the adjacency matrix with self-connections. When $k = 0$, $H_0 = X$ is the original graph's node features, $\tilde{D}$ is the degree matrix for graph G, i.e., a diagonal matrix containing the number of edges connected to each node. W^k is a trainable weight matrix, and $\tilde{D}^{-(1/2)}\tilde{A}\tilde{D}^{-(1/2)}$ is to normalize matrix A and obtain a symmetric one.

In our module, multiple graph convolution layers are stacked [see Fig. 4(a)] to learn multiple sets of node embeddings $\{Z^1, Z^2, \ldots, Z^n\}$. For each graph convolution layer l, $Z^{(l)} = \mathrm{GCN}(A^{(l)}, X^{(l)})$ yields the embedding matrix. Here $A^{(l)}, X^{(l)}$ are the adjacency matrix and node feature matrix, respectively. $X^{(0)}$ is the one-hot coding of object labels as the initial node features. Then, the merged $Z^{(l)}$ can be written as follows:

$$Z^{(l)} = \sum_{i=1}^{n}\left(\alpha_i \otimes GCN_{i,\mathrm{embed}}\left(A^{(l)}, X^{(l)}, d_i\right)\right) \tag{3}$$

where d_i is the output node feature dimension, and α_i, a trainable weight matrix, represents the weight of different node embeddings. $\sum$ indicates the concatenation operation of n tensors.

b) Hierarchical graph pooling: After the convolution, we perform the following pooling operation to shrink the graph. Compared with global pooling methods (such as sum, mean, and max pooling), hierarchical graph pooling can better retain the local information of the graph structure.

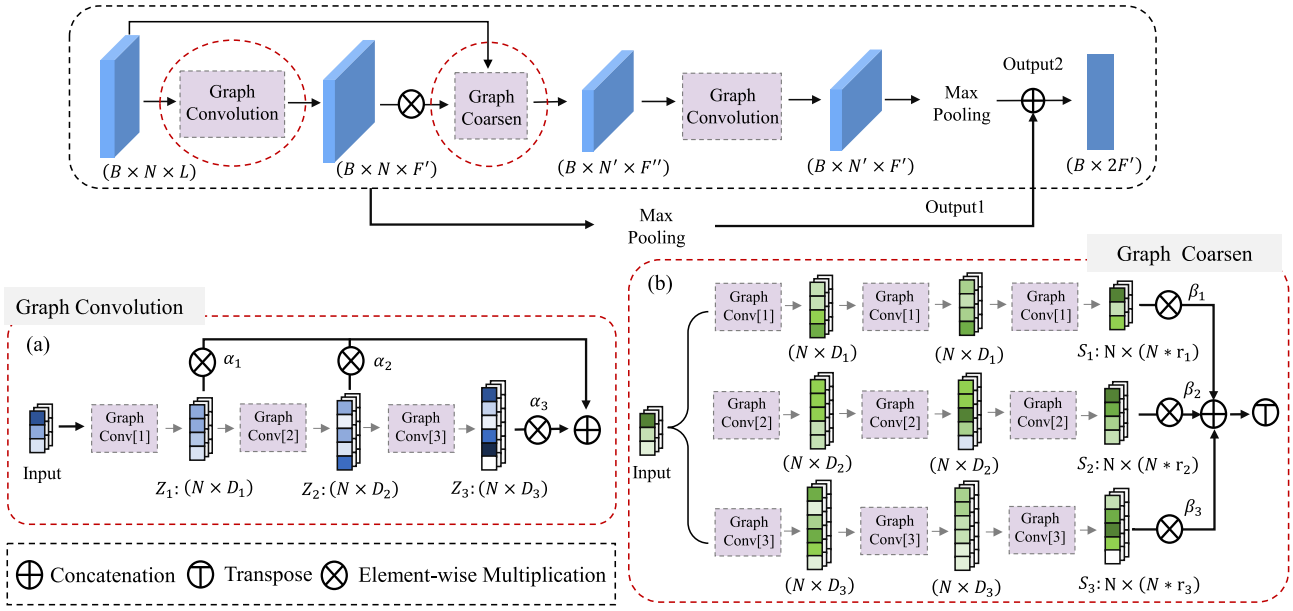


Fig. 4. Detail of graph convolution and graph coarsen module. (a) Graph convolution module, (b) Graph coarsen module.

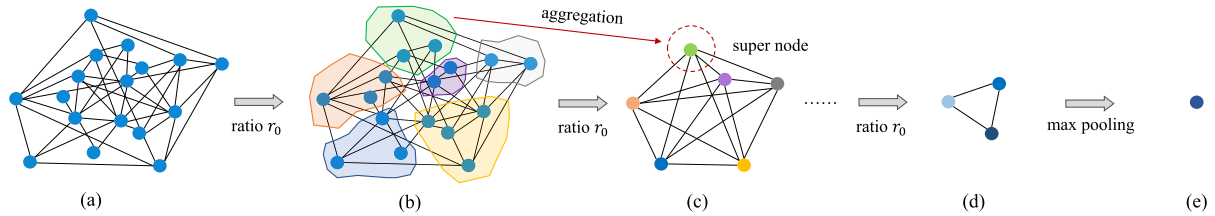


Fig. 5. Diagram of hierarchical graph pooling. (a) Indoor scene graph G_0, (b) Assigned subgraphs, (c) Coarsened graph 1, (d) Coarsened graph n, and (e) Graph representation.

Here, an assigned matrix method, MxPool [14], is selected as our hierarchical graph pooling operator. It divides a graph into subgraphs by ratios, and each subgraph can be aggregated as a super node. These nodes form a new coarsened graph with full connections, which is fed to the next GCN layer.

Fig. 5 shows the process of coarsening graphs: Given an indoor scene graph G_0 with node features X_0, adjacency matrix A_0 and coarsen ratio r_0, we learn node embeddings and downsample graphs with ratios using a GCN, until there is only one node left, that is, the final graph representation.

The coarsening ratio should match the size of the graph. In other words, a small ratio should be used for the small graphs; therefore, to find the best coarsening ratio for each graph, define the assignment matrix $S^{(l)} = \mathrm{GCN}(A^{(l)}, X^{(l)})$ and use multiple GCNs [see Fig. 4(b)] to learn n assignment matrices $\{S^1, S^2, \ldots, S^n\}$. Then the merged $S^{(l)}$ is as follows:

$$S^{(l)} = \sum_{i=1}^{n} \left(\beta_i \otimes \mathrm{GCN}_{i,\mathrm{pool}}\left(A^{(l)}, X^{(l)}, r_i\right)\right) \tag{4}$$

where r_i is the coarsening ratio of graph G_i, and β_i represents the weight of different assignment matrices. $\sum$ indicates the concatenation operation of n tensors.

Finally, given the merged node embeddings $Z^{(l)}$ of (3) and merged assignment matrix $S^{(l)}$ of (4) as inputs, the adjacency matrix $A^{(l+1)}$ and cluster embeddings $X^{(l+1)}$ of the new coarsened graph are generated by the two following equations [47]:

$$X^{(l+1)} = S^{(l)^T} Z^{(l)} \tag{5}$$

$$A^{(l++1)} = S^{(l)^T} A^{(l)} S^{(l)}. \tag{6}$$

Note that the coarsened graph is fully connected [such as Fig. 5(b) and (c)], and its size is the number of n clusters, namely, one cluster corresponds to one super node in the new graph. Where $A^{(l+1)}$ are a real matrix and each entry in $A^{(l+1)}$ denotes the edge weight between two clusters. Then the cluster embeddings $X^{(l+1)}$ and the coarsened adjacency matrix $A^{(l+1)}$ will be entered in to the next GCN layer $l+1$. The above graph convolution and graph pooling process is performed until a graph representation is obtained.

2) Object Attention-Based Global Feature Aggregation Module: In addition to learning spatial relations between objects (local information), global information is required to improve scene representation ability and recognition accuracy; therefore, the OA-GFAM is proposed. Each node's embeddings are updated by aggregating other nodes' features based on object attention scores. Then, the graph representation is generated by the sum pooling operation, which describes the global information of the indoor scene. The process of global information aggregation is shown in Fig. 6.

In this module, we focus on learning the relative importance score (called "object attention score") for each node by the global attention mechanism, which describes the weight of the

TABLE I
STATISTICS OF INDOOR POINT CLOUD DATASET

Dataset	Scans	Object Classes	Scene Classes	Max Objects	Min Objects	Avg. Objects
ScanNet	1513	20	21	132	4	32.14

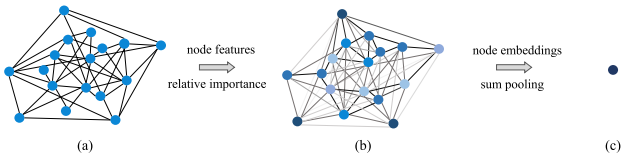


Fig. 6. Diagram of global information aggregation. (a) Indoor scene graph G_0, (b) Updated node embeddings, and (c) Graph representation.

central node relative to others. To this end, Self-attention [66] is used to capture attention scores between all nodes and to update node embeddings. Specifically, we introduce two simple but effective designs of encoding to help learn the graph representation well (see Fig. 7). We elaborate on these designs in the following:

a) Self-attention module: Self-attention mechanism is first proposed by Vaswani et al. [66]. In this module, the input node features X are first projected to query (Q), key (K), and value (V) matrices through a linear projection, respectively. The self-attention is then calculated as follows:

$$\text{Output}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{7}$$

where Q, K, V, and output are all vectors and d_k is the dimension of keys, a scale parameter. Q, K, and V are first initialized and assigned through different linear transformations. After that, Q and K are multiplied together to calculate the correlation matrix of each node, and attention scores are generated by the softmax function.

b) Quantity encoding: In (7), the attention score is usually calculated according to the semantic correlation between object nodes. The object number of each class, which measures how important these nodes are in the graph, is, however, a strong signal for graph understanding. For example, if there are twenty objects in a scene, including twelve chairs, two doors, one ceiling, one floor, and four walls, the chair plays an important role in this scene. To this end, the object number is introduced as an additional signal to the neural network.

We design a *quantity encoding*, which assigns each node two real-valued embedding vectors according to the object number of each class and their proportion in all. As the quality encoding is applied to each node, we simply add it to the node features as the input

$$X_i^0 = x_i + n_{c_{v_i}} + p_{c_{v_i}} \tag{8}$$

where x_i represents initial node features, c is the object class of node v_i, n, and p are the object number of this class and its proportion in all objects, respectively.

c) Spatial position encoding: The advantage of self-attention is its global receptive field; however, it hard to capture spatial features among objects. In fact, the different distribution of objects in indoor space is a key factor to distinguish similar scenes. Such information is neglected in the current global attention calculation. And we believe it should be a valuable signal.

We, therefore, present *spatial position encoding*, which measures the spatial correlation between two objects. The closer they are, the more relevant they are. Concretely, for a graph G_i, a function $r_{(v_i,v_j)}$ is considered to present the spatial correlation between node v_i and v_j. In this article, the function r is defined by the reciprocal of spatial distance between two nodes in the graph. We incorporate *spatial position encoding* via a bias term to the attention module

$$A_{(i,j)} = \frac{Q_i K_i^T}{\sqrt{d_k}} + r_{(v_i,v_j)}$$
$$r_{(v_i,v_j)} = \frac{1}{s(v_i,v_j) + \varepsilon} \tag{9}$$

where $A_{(i,j)}, s$ are the attention score and the spatial distance of node v_i and v_j. ε is a minimal scalar to avoid zero value.

IV. EXPERIMENTS AND ANALYSIS

A. Dataset

The proposed method is tested on a public indoor 3-D point cloud dataset, ScanNet v2 [57], which is the most accepted and robust dataset in indoor 3-D scene recognition. ScanNet v2 is an RGB-D dataset containing 1513 scans. It has 21 scene classes and 20 object classes and includes hundreds to thousands of object instances already detected in point clouds with semantics. Statistics of the dataset are summarized in Table I. Figs. 8 and 9 show the point clouds for different scene types and corresponding sample numbers.

B. Experiment Setting

1) Experiment Environment: The proposed indoor graph classification network is implemented in Python language and the Pytorch library [58] with a single RTX 2070Ti GPU. We train the network for 300 epochs using Adam Optimizer [59] with a learning rate of 0.0001 and set the batch size to 64 for the ScanNet v2 dataset. The loss function is Cross Entropy.

2) Accuracy Evaluation Method: Two evaluation methods are adopted in the experiments, namely fixed training and validation dataset and ten-fold cross validation. In experiments, we use the same training and validation datasets as in [6] to evaluate, compare, and test our method. In particular, 1013 scans for training and 500 scans for validation. The relevant results are shown in Tables II, III, IV, VI, VII, VIII, IX, and X.

The ten-fold cross-validation method is only adopted to illustrate the average performance of our model and compare it with existing methods objectively. The results are reported

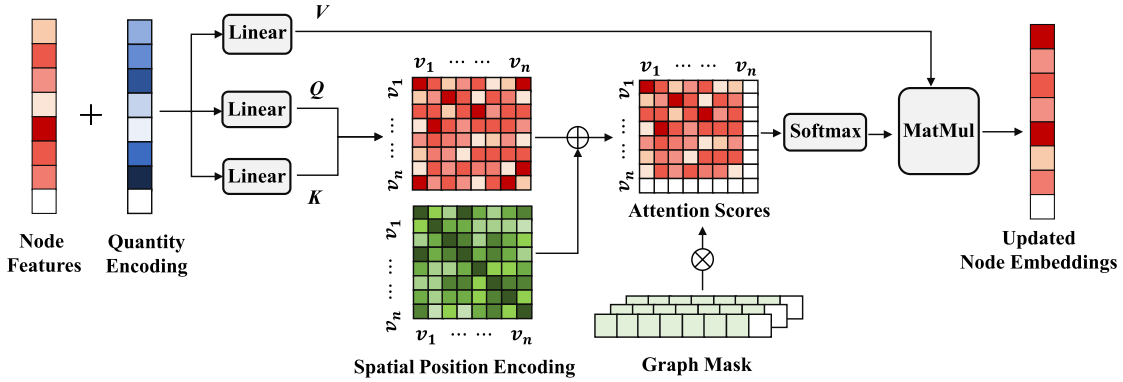


Fig. 7. Diagram of updating node embeddings based on object attention scores.

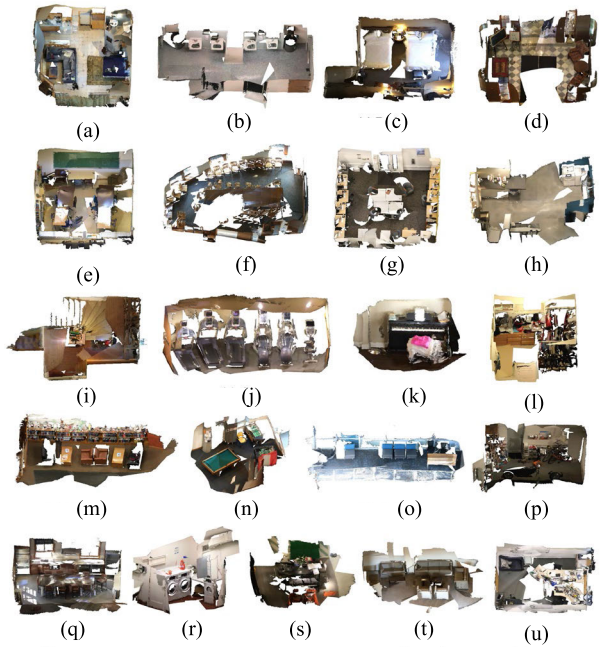


Fig. 8. Typical point clouds of 21 indoor scene types. (a) Apartment, (b) Bathroom, (c) Bedroom, (d) Living room, (e) Classroom, (f) Conference room, (g) Computer cluster, (h) Copy room, (i) Stairs, (j) Gym, (k) Misc room, (l) Closet, (m) Bookstore, (n) Game room, (o) Hallway, (p) Storage, (q) Kitchen, (r) Laundry room, (s) Dining room, (t) Lobby, and (u) Office.

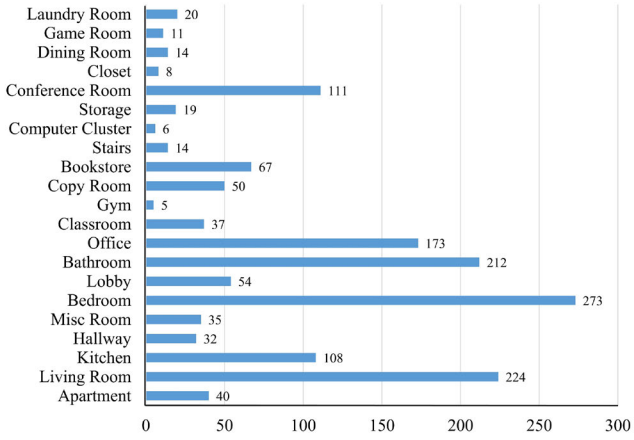


Fig. 9. Sample distribution of 21 scene classes in ScanNet v2.

TABLE II
RESULTS OF VARYING DIFFERENT BUFFER DISTANCE (%)

Distance	0.00m	0.10m	0.20m	0.25m	0.50m
Accuracy	85.00	85.80	88.00	86.00	84.00
Precision	76.11	82.15	83.71	78.32	73.05
Recall	72.34	72.60	80.94	76.00	73.32
F1-score	74.18	77.08	82.30	77.14	73.18

TABLE III
RESULTS OF VARYING DIFFERENT FEATURE DIMENSION (%)

Dimension	10	20	30	50	80
Small	71.80	75.19	**80.82**	**82.71**	78.95
Medium	85.42	89.29	**89.52**	**89.81**	86.42
Large	83.46	85.19	87.31	**89.58**	**91.34**

TABLE IV
RESULTS OF VARYING DIFFERENT COARSEN RATIO (%)

Ratio	0.05	0.10	0.15	0.20	0.50
Small	80.45	**82.71**	80.82	**81.58**	79.70
Medium	89.42	**89.81**	**89.52**	86.15	89.23
Large	87.50	89.58	**91.37**	**91.67**	81.25

in Table VII. The dataset is split into two parts: 80% as a training set and the remaining 20% as a validation set. This random splitting process is repeated 10 times.

In this article, four evaluation metrics are used to evaluate the performance of classification. They are accuracy, precision, recall, and F1 score. The primary metric used on the ScanNet v2 dataset is the accuracy [see (10)] of the model. It is a good representation to describe the overall performance of the classifier. The precision, recall, and F1 score [see (13)] are, moreover, calculated for each indoor scene class. Precision and Recall are calculated using (11) and (12), respectively, using the true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{10}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{11}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{12}$$

$$F1 - \text{score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{13}$$

3) Parameter Selection: In our experiment, we set the buffer distance to 0.2 m to generate indoor scene graphs. The dimensions of Q, K, and V are the same as 64 by linear transformations of input node features in the OA-GFAM branch. In the SR-LFAM module, three GCN layers are stacked to learn node embeddings with different dimensions, and three GCN layers are used to learn assignment matrices with different pooling ratios. To be specific, the dimensions of node representations are set to [30,50,80], and the coarsening ratios are set to [0.1,0.15,0.2]. The reason why these hyperparameters are selected is given in the following.

a) Selection of buffer distance: In the graph construction algorithm (ISGC), the main principle is to find neighbors around the object by buffer distance and then establish edge connections to form an indoor scene graph. Table II shows the impact of different distance settings on the result, and the optimal buffer distance selected in this article is 0.2 m.

b) Selection of feature dimension and coarsening ratio: Previous research has shown that different sizes of graphs are suitable for different node feature dimensions and coarsening ratios. For indoor scene graph classification, we select multiple node dimensions [30,50,80] and graph coarsening ratios [0.1,0.15,0.2] through two comparative experiments.

Specifically, according to the node number contained in the graph, the indoor scene graphs are divided to small graphs [0, 20], medium graphs [21, 35], and large graphs [36, 132]. In the first experiment, we fixed the coarsening ratio to 0.1 and then changed the node feature dimension to observe the result (see Table III). In the second experiment, we set the node dimension to 50 and change the coarsening ratio, the comparison result is shown in Table IV.

C. Indoor Scene Graphs Construction

Our proposed method is based on indoor scene graphs, which are the input for the graph classification network. Indoor 3-D point clouds are, therefore, preprocessed to a set of scene graphs by the ISGC method. We set the buffer distance of the 3-D bounding box to 0.2 m.

After that, 1513 scans of the ScanNet v2 dataset are formed into corresponding 1513 graphs. Each graph represents a scene containing object nodes and edges. The details of these graphs are provided in Table V. Fig. 10 shows the comparison between point clouds and graphs for some scene types. From this figure, we can see that the ISGC method can accurately convert an indoor 3-D scene from point clouds to a graph by abstracting objects into nodes and representing the spatial relationship between objects through edges. For example, as shown in Fig. 10(c), there are floors, doors, walls, windows, and tables surrounded by chairs in this classroom. Our constructed graphs can distinguish these objects and detect their neighbors to represent spatial layout by edges.

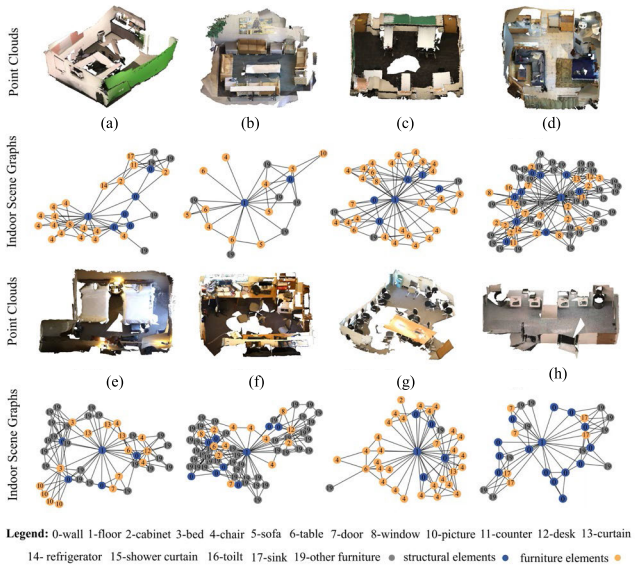


Fig. 10. Constructed indoor graph examples of different scenes. (a) Kitchen, (b) Living room, (c) Classroom, (d) Apartment, (e) Bedroom, (f) Office, (g) Conference room, and (h) Bathroom.

D. Indoor Graph Classification Performance

In this subsection, we give the performance of the proposed method and select accuracy, precision, recall, and F1 score to evaluate the results in two validation methods.The proposed method can achieve the task of indoor scene recognition, which further shows the effectiveness of our ISGC method and the classification network.

The ten-fold cross validation is adopted to objectively report the average performance of our model. We obtain an overall accuracy of 90.46%. The average precision, recall, and F1 score are 82.91%, 80.04%, and 81.45%, respectively.

For the fixed training and validation dataset, our model obtains 88.00% accuracy, 83.71% precision, 80.94% recall, and 82.30% F1 score. Table VI reports the evaluation result in detail, and the first row shows the number of samples for each scene. According to this table, there are the following observations:

1) As shown in Table VI, all 21 indoor scene types are detected by our proposed method. From this table, it can be seen that 12 scene classes are well recognized and their F1 score values are greater than 80%; moreover, among them, there are eight scene classes, namely kitchen, bedroom, copy room, bathroom, dining room, conference room, office room, and game room, where their F1 scores are more than 90%.

2) Some scene types, like hallway and music room, however, cannot be classified effectively; here, the F1 scores are lower than 60%. Compared with sample numbers, we can see that the sample numbers of these two scene types are 32 and 35, respectively. They are smaller than office room, conference room, bedroom, etc. The gym, however, has only five samples, and it can also be detected with a high F1 score of 80.00%. We can, therefore, observe that these scene graphs do not have specific objects and spatial structures, similar to other classes. In other words, they are hard samples, which cannot be learned easily in the training process.

TABLE V
STATISTICS OF CONSTRUCTED INDOOR GRAPHS

Dataset	Graphs	Node Classes	Scene Classes	Max Nodes	Avg. Nodes	Avg. Edges
ScanNet	1513	20	21	132	32.14	97.87

TABLE VI
EVALUATION RESULTS FOR 21 INDOOR SCENE TYPES

	Apartment	Living room	Kitchen	Hallway	Msic room	Bedroom	Lobby
Number	40	224	108	32	35	273	54
Precision	90.91	82.05	100.00	40.00	66.67	93.26	73.33
Recall	62.50	88.89	94.29	75.00	47.06	98.81	52.38
F1-score	74.07	85.33	**97.06**	52.17	55.17	**95.95**	61.11
	Copy room	Storage	Stairs	Computer cluster	Bookstore	Bathroom	Closet
Number	50	9	14	6	67	212	8
Precision	100.00	66.67	80.00	50.00	82.76	95.95	100.00
Recall	88.89	66.67	100.00	100.00	88.89	95.95	50.00
F1-score	**94.12**	66.67	88.89	66.67	85.71	**95.95**	66.67
	Dining room	Laundry room	Classroom	Conference room	Game room	Office	Gym
Number	14	20	37	111	11	173	5
Precision	100.00	80.00	70.00	92.31	100.00	94.00	100.00
Recall	100.00	57.14	77.78	94.74	100.00	94.00	66.67
F1-score	**100.00**	66.67	73.68	**93.51**	**100.00**	**94.00**	80.00

TABLE VII
COMPARED RESULTS OF DIFFERENT METHODS

	Method	Acc_1 [%]	Acc_2 [%]	Param [M]
Point-based	Pointnet [28]	70.80	---	0.68
	DGCNN [29]	77.20	---	1.80
Voxel-based	Resnet14 [40]	87.40	---	21.50
	Multi-task [6]	90.30	---	35.30
Object-based	BoW+DT	74.60	74.80	---
	BoW+RF	83.20	86.28	---
	BoW+SVM	76.10	72.44	---
	Ours	88.00	90.46	0.21
	Ours w/o QE	86.20	87.95	0.21
	Ours w/o SPE	87.00	88.54	0.21
	GCN [41]	77.80	76.75	0.31×10^{-2}
	Diffpool [47]	83.20	82.90	0.20×10^{-1}
	MxPool [14]	85.40	86.85	0.19
	SAT [64]	36.60	40.19	0.67
	GraphGPS [65]	68.90	70.14	0.60
	GT [63]	83.60	85.11	0.59

TABLE VIII
COMPARISON OF DIFFERENT INDOOR GRAPH CONSTRUCTION (%)

Method	K-NN	Fully-connected	Ours
Accuracy	84.00	83.60	88.00
Precision	74.55	76.70	83.71
Recall	72.61	68.66	80.94
F1-score	73.57	72.35	82.30

TABLE IX
RESULTS OF VARYING DIFFERENT DROPPING RATE (%)

Ratio	0%	5%	10%	20%	30%	40%
Accuracy	88.00	84.20	81.20	74.60	67.67	58.20
Precision	83.71	76.83	74.19	59.92	51.32	41.57
Recall	80.94	74.55	71.22	59.76	56.38	42.50
F1-score	82.30	75.67	72.67	59.84	53.73	42.03

TABLE X
COMPARED RESULTS OF USING STRUCTURAL ELEMENTS (%)

Index	Proposed method	
	without structural elements	with structural elements
Accuracy	83.60	88.00
Precision	73.75	83.71
Recall	71.48	80.94
F1-score	72.60	82.30

E. Comparisons

In this subsection, we compare previous methods for the indoor 3-D scene recognition task, including point-based methods (PointNet [28] and DGCNN [29]) and voxel-based methods (Resnet [40] and multitask [6]), as mentioned in the introduction. Specifically, the point-based method works by dividing original point clouds into blocks as input and then learning local or global features per point in different ways. The voxel-based method feeds sparse voxel grids derived from point clouds and then obtains features through sparse convolution. In addition, we also compare our method against the Bag of Words (BoW) model [62] and various graph classification networks (like GCN [41], DiffPool [47], MxPool [14], SAT [64] GraphGPS [65] and Graph Transformer [63]). These methods use indoor objects identified from raw point clouds, which can be named an object-based method.

Here we use the BoW model combined with machine learning methods to classify indoor scenes. For our indoor

recognition task, for a scene, ignoring the spatial relationship between objects and counting the object number of each type in the room, a BoW model of the indoor scene can be built. Then using machine learning methods such as SVM, Decision Tree (DT), or Random Forest (RF) to do scene classification.

The overall accuracy and model parameters compared for 21 scene types are displayed in Table VII. Two results are reported. One is based on a fixed training and validation dataset, represented by Acc_1. In this experiment, our method is compared with all methods (point-based methods, voxel-based methods, and object-based methods). The other is the result of ten-fold cross validation, represented by Acc_2. We only compare our method with object-based methods. The reason is that point-based and voxel-based methods are not suitable for ten-fold cross validation. The input of these methods is millions of points. Hence, they have extremely high time complexity.

From this table, it can be seen that our method is more computationally efficient than the methods that use point or voxel embeddings and better in terms of accuracy than the BoW and graph classification methods. Concretely, our method achieves much better results, with 88.0% overall accuracy, than point-based methods and Resnet. Although the accuracy of our model is lower than the voxel-based multitask method, it only has 0.21 million parameters, which is 100 times smaller than that of the multitask method. This means that our model can save a lot of memory space during computation.

Compared to BoW methods, our method is the best one in accuracy because we use spatial relationships to describe the scene rather than only object semantics and numbers. It indicates spatial information is an important factor in scene recognition. The comparison of graph classification networks is also shown in Table VII. Our method is superior than others, regardless of the accuracy evaluation method used.

We also conducted ablation experiments to demonstrate the impact of two encodings. In terms of Acc_1, the accuracy will decrease by 1.80%, 1.00% without using quantity encoding and spatial position encoding, respectively.

F. Analysis

1) Impact of Different Scene Graph Construction: In this subsection, the effectiveness of the proposed indoor graph construction (ISGC) method is tested. Two other algorithms namely K-NN and fully connected are selected to construct different indoor scene graphs.

K-NN finds a set of k nearest neighbors for each node in Euclidean space, which is a very common algorithm to describe local spatial relationships. To create indoor scene graphs, each object node is taken as the center to search for the nearest k neighbor points to establish edges with the central node. Here, the parameter k is set to 9, which is the best empirical value through many experiments. While the fully connected method is that each node is connected to its remaining nodes, which preserves the relationship between all nodes in the graph.

In this experiment, in order to make a fair comparison, we set the same model parameters and used the fixed training and validation dataset to learn graph representation for different graph structures. Fig. 11 vividly shows indoor scene graphs constructed by different algorithms for the same indoor scene. The compared quantitative results of scene recognition are shown in Table VIII. From this result, we find that our ISGC method has achieved the best performance on four evaluation metrics with 88.00% accuracy, 83.71% precision, 80.94% recall, and 82.30% F1 score, which shows the effectiveness of our ISGC.

2) Impact of Missing Indoor Objects: In previous experiments, ScanNet v2 is used, which is a perfect dataset. It means all indoor objects are detected successfully in 3-D point clouds. In practical applications, however, robots cannot completely and accurately identify all objects; for example, dynamic objects (person, etc.) inevitably bring occlusion in point clouds; therefore, in this subsection, we simulate the actual situation manually, that is, randomly dropout 5%, 10%, 20%, 30%, and 40% indoor objects in the validation set, and then observe their changes in results. These compared results are reported in Table IX.

It is expected that the average classification accuracy degrades due to information loss caused by increased occlusion, i.e., having fewer detected indoor objects. When 5% of objects are discarded, the accuracy decreases by no more than 5%. With a drop rate of 30%, the accuracy is, moreover, almost similarly affected by 20.33%. This result shows that the scene classification accuracy depends sublinearly on the rate of dropped objects in the scene. It also means our method is robust to common scanning artifacts such as occlusion and noise, and to common instance segmentation artifacts such as class imbalance. The average precision, recall, and F1 score are detailed in Table IX.

3) Impact of Indoor Specific Objects: The type of indoor scene is related to interior objects and their spatial structures. Specific objects may play a key role in indoor scene recognition. For example, a room with beds is likely to be a bedroom or an apartment, bathtubs belong to the bathroom, and bookshelves to bookstores. To check the influence of specific objects on scene types, object classes are removed from the validation dataset in turn, and then the change in results is observed. Here, the removed object classes are only furniture elements (such as chairs, tables, beds, etc.), because structural elements are common in each indoor scene type and there is a subset of 15 classes given (scene classes with less than five samples in the validation have been removed). The reason for this is that the number of samples is too small to be universal, resulting in extreme values.

The change in the recall is shown in Fig. 12. From this figure, it is obvious that there are pronounced drops for the bedroom when removing the beds, and for the conference room and classroom when removing chairs. On the contrary, the laundry room class benefits greatly from removing toilets or sinks. In the following, we give the confusion matrices after removing beds, chairs, and toilets, respectively, and compare them with the original results.

Fig. 13(a) is the initial result, and Fig. 13(b) is the matrix with removing bed nodes. The red rectangles mark the comparison of the bedroom. It is obvious that when all bed nodes

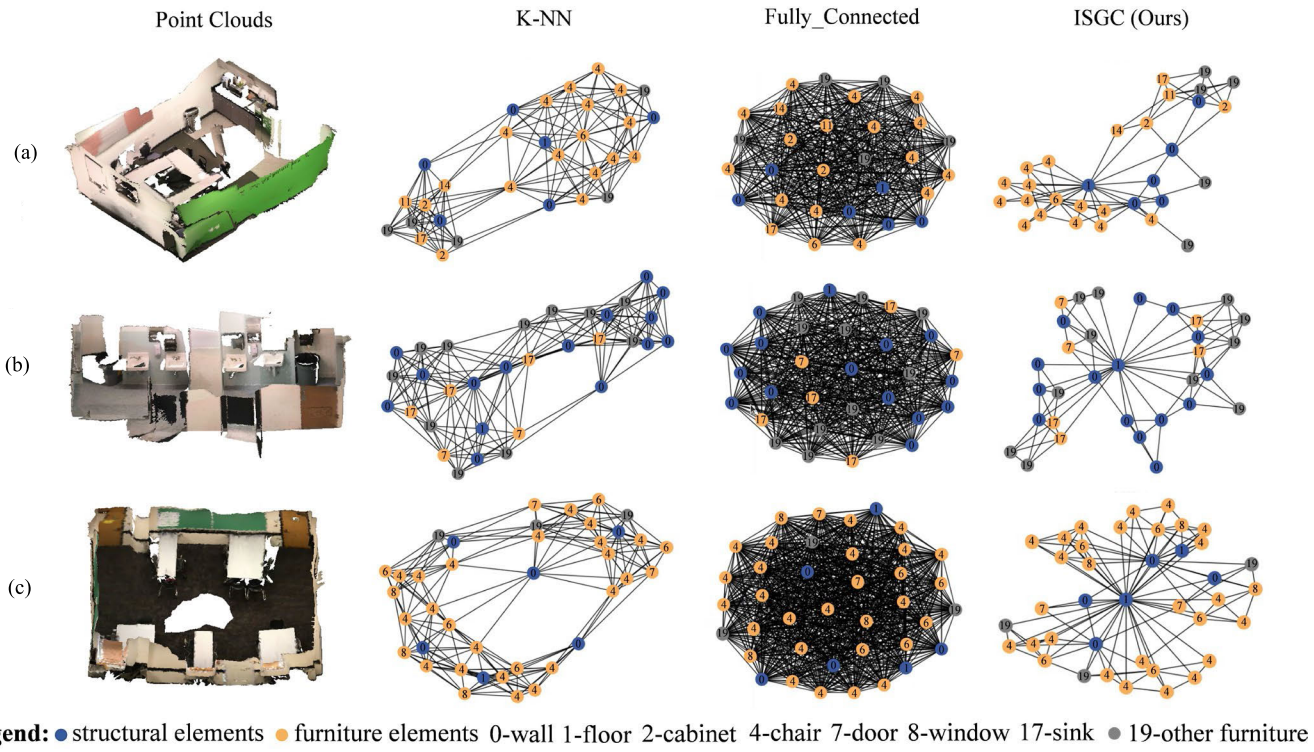


Fig. 11. Indoor graphs' visualization of different algorithms. (a) Kitchen, (b) Bathroom, and (c) Classroom.

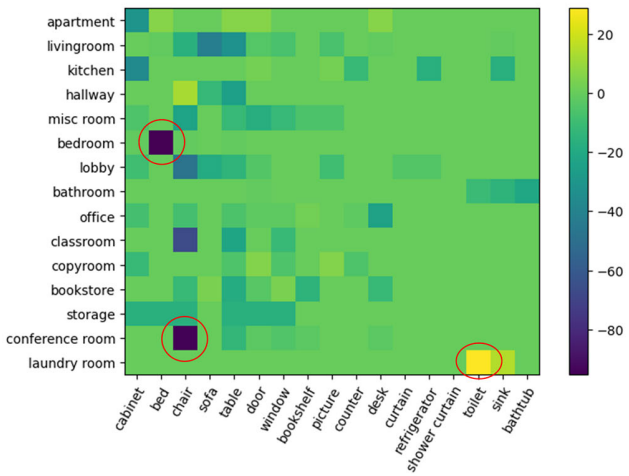


Fig. 12. Change of recall per scene type after removing all nodes with a certain object class.

are removed from the validation set, the classification results of the bedroom will be greatly affected. Most of them are mistakenly divided into an office and storage. From Fig. 13(c), we find that when removing the chair nodes, the result of the conference room changed significantly (circled by green rectangles). All conference rooms cannot be recognized and are, therefore, classified as a living room, hallway, copy room, etc., incorrectly. It illustrates the importance of specific objects in the scene recognition task. Beds and chairs appear to be the key objects for bedrooms and conference rooms, respectively.

What surprised us is that when we removed the toilet nodes, the accuracy of the laundry room, however, improved. The yellow rectangles mark the comparison results of the laundry room, as shown in Fig. 13(a) and (d). Before removing the toilets, the laundry room is often mistaken for the bathroom, but now it is distinguished. Seemingly, this prevents confusion with other scene types. We observed the object distribution of the laundry room in the training and validation dataset and found that there are no toilet objects in all the training sets and only two samples with toilets in the validation set; therefore, when removing the toilets, these two samples can be classified successfully, which leads to increased accuracy.

In addition, from Fig. 13, most scenes are affected by three or more object classes, such as living room, lobby, storage, etc., and bookshelves and bathtubs will not seriously affect the classification results of bookstores and bathrooms, which is different from our previous expectations. It also illustrates that indoor scene recognition is not only the search for specific objects but also exploits more complex patterns such as co-occurrence and spatial layout of objects.

4) Impact of Indoor Structural Elements: As mentioned in Section III-B, the interior objects contained in a scene can be divided into structural elements (ceiling, floor, wall, etc.) and furniture elements (chair, table, sofa, bookcase, door, window, etc.). They are key objects to preserving the indoor spatial structure completeness.

Structural elements are common in each scene, which may bring noises for our task. Hence, we added an additional experiment to illustrate the impact of structural elements on the result. Concretely, we reconstruct the indoor scene graphs, removing structural elements and their edge connections. To get a fair comparison, the fixed 1013 scans for training and 500 scans for validation are used in this

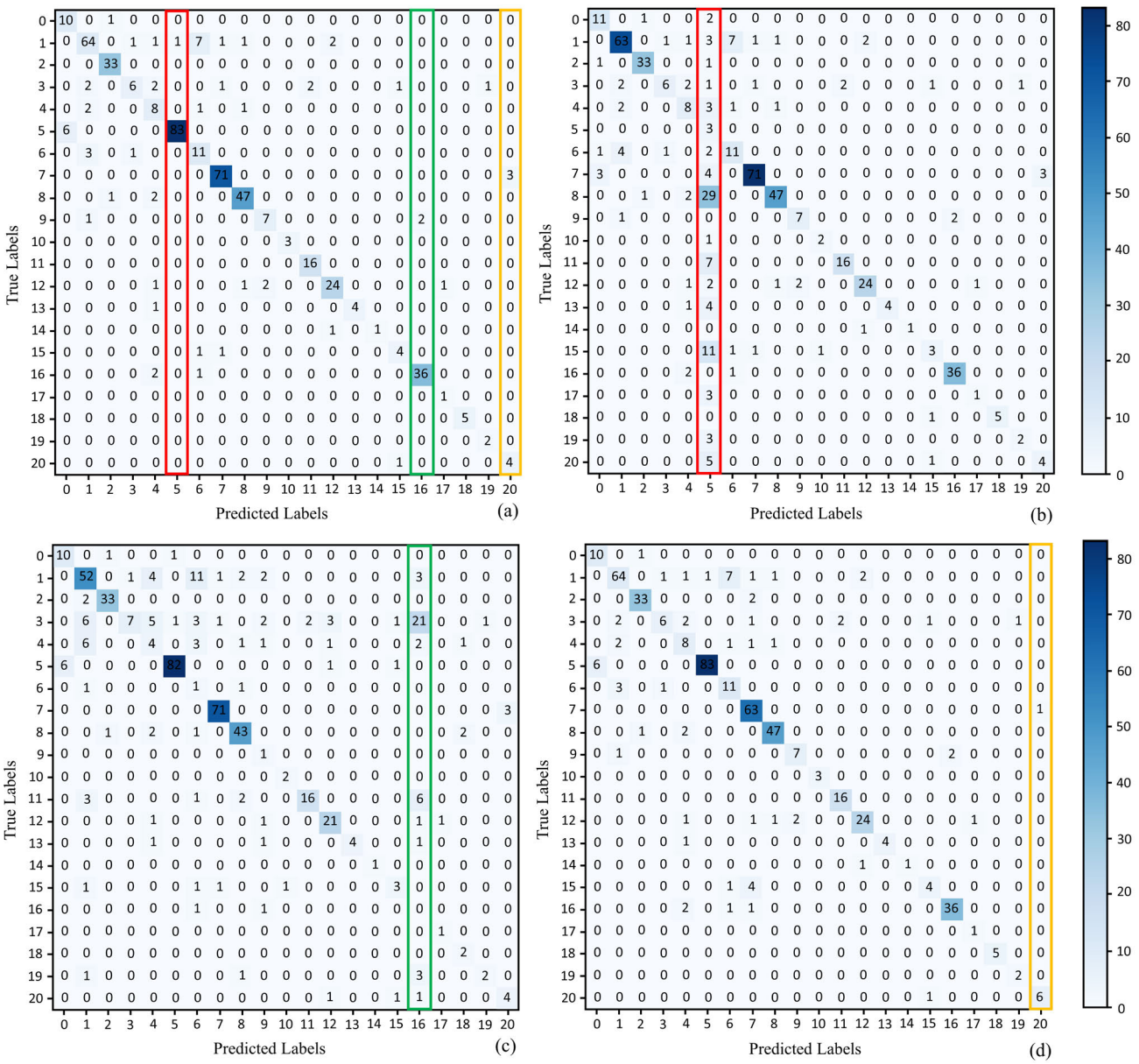


Fig. 13. Confusion matrices showing object counts. (a) Initial result is without dropping any objects. (b) Beds removed. (c) Chairs removed. (d) Toilets removed.

experiment, which is consistent with our method. The comparison result of four metrics is shown in Table X. From this table, the accuracy decreases by 4.80% if we do not use structural elements, which indicates they are necessary elements in our constructed indoor scene graphs.

V. Conclusion

Scene recognition is a fundamental task in 3-D scene understanding. Aiming at the problems existing in the current methods, namely computational efficiency, spatial structure completeness, and decoration style consistency, we present a new approach for indoor scene recognition based on a set of 3-D scene graphs to balance the above shortcomings. These graphs are constructed from 3-D objects derived from indoor point clouds, where objects are considered as nodes and edges describe the spatial relationships between the nodes. Then, we present a novel indoor scene classification network to learn different graph representations at local and global levels. It includes SR-LFAM and OA-GFAM.

Experiments with the ScanNet v2 point cloud dataset yield good results with up to 88.00% accuracy and 82.30% F1 score in the fixed validation dataset and 90.46% accuracy and 81.45% F1 score in the ten-fold cross validation method; moreover, we analyzed the robustness of the proposed method thoroughly. Increasing the percentage of random missing objects in the input data reduces the F1 score with a close to linear dependence, indicating sufficient robustness against random incompleteness in data (see Section IV-F2). Picking out specific labeled objects from the data yields expected results with the method not being highly sensitive to the removal, except in a few understandable cases, e.g., bed and toilet (see Section IV-F3). This illustrates that the proposed

method not only searches for specific objects but also exploits more complex patterns such as co-occurrence and spatial layout of objects. It is sufficiently robust against noise of input data.

Compared to the state-of-the-art (SOTA), our method significantly reduces the number of parameters for training and inference (see Table IV), which leads to more computational efficiency than what is offered by the current point-based and voxel-based deep learning methods. Related to this, we are quite confident that our model architecture can handle even larger datasets than ScanNet v2 because the approach that inspired us has been originally tested on big protein data [14]. In addition, our method offers higher classification accuracy than the traditional BoW-based classifiers and other graph classification networks, because it leverages embeddings of object semantic, quantity and spatial relationship.

In the future, scene graphs are likely to become more common and more complex, and the proposed method will likely gain more importance. Our approach is based on indoor scene graphs composed of object instances, and therefore, its result is affected by these objects detected from point clouds by instance segmentation. If a large number of objects are undetected or identified incorrectly, it will lead to a decrease in performance. Additionally, our model has not been tested in various indoor large-scale scenarios, which is another limitation and is also our future work to focus on.

REFERENCES

[1] M. Narasimhan et al., "Seeing the un-scene: Learning amodal semantic maps for room navigation," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Jul. 2020, pp. 513–529.

[2] D. Chaplot, D. Gandhi, A. Gupta, and R. Salakhutdinov, "Object goal navigation using goal-oriented semantic exploration," in *Proc. NIPS*, Jul. 2020, pp. 4247–4258.

[3] V. V. Lehtola, S. Nikoohemat, and A. Nüchter, "Indoor 3D: Overview on scanning and reconstruction methods," in *Handbook of Big Geospatial Data*. Springer, 2020, ch. 3, pp. 55–97.

[4] P. Espinace, T. Kollar, A. Soto, and N. Roy, "Indoor scene recognition through object detection," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2010, pp. 1406–1413.

[5] B. Miao, L. Zhou, A. S. Mian, T. L. Lam, and Y. Xu, "Object-to-scene: Learning to transfer object knowledge to indoor scene recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2021, pp. 2069–2075.

[6] S. Huang, M. Usvyatsov, and K. Schindler, "Indoor scene recognition in 3D," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2020, pp. 8041–8048.

[7] J. Johnson et al., "Image retrieval using scene graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, Jun. 2015, pp. 3668–3678.

[8] J. Johnson, A. Gupta, and F. Li, "Image generation from scene graphs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, Apr. 2018, pp. 1219–1228.

[9] S. Nikoohemat, A. Diakite, S. Zlatanova, and G. Vosselman, "Indoor 3D modeling and flexible space subdivision from point clouds," *ISPRS Ann. Photogramm., Remote Sens. Spatial Inf. Sci.*, vol. 4, pp. 285–292, May 2019.

[10] I. Armeni et al., "3D scene graph: A structure for unified semantics, 3D space, and camera," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5663–5672.

[11] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone, "3D dynamic scene graphs: Actionable spatial perception with places, objects, and humans," in *Proc. IEEE/CVF Conf. Robot. Sci. Syst. (RSS)*, Feb. 2020, pp. 1–11.

[12] J. Wald, N. Navab, and F. Tombari, "Learning 3D semantic scene graphs with instance embeddings," *Int. J. Comput. Vis.*, vol. 130, no. 3, pp. 630–651, Jan. 2022.

[13] C. Zhang, J. Yu, Y. Song, and W. Cai, "Exploiting edge-oriented reasoning for 3D point-based scene graph analysis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, Jun. 2021, pp. 9700–9710.

[14] Y. Liang, Y. Zhang, D. Gao, and Q. Xu, "MxPool: Multiplex pooling for hierarchical graph representation learning," 2020, *arXiv:004.06846*.

[15] A. Quattoni and A. Torralba, "Recognizing indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 413–420.

[16] F. Liu et al., "RemoteCLIP: A vision language foundation model for remote sensing," 2023, *arXiv:2306.11029*.

[17] N. Sun, W. Li, J. Liu, G. Han, and C. Wu, "Fusing object semantics and deep appearance features for scene recognition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 6, pp. 1715–1728, Jun. 2019.

[18] X. Song, S. Jiang, B. Wang, C. Chen, and G. Chen, "Image representations with spatial object-to-object relations for RGB-D scene recognition," *IEEE Trans. Image Process.*, vol. 29, pp. 525–537, 2020.

[19] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, Jun. 2006, pp. 2169–2178.

[20] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba, "SUN database: Large-scale scene recognition from abbey to zoo," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, Jun. 2010, pp. 3485–3492.

[21] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 6, pp. 1452–1464, Jun. 2017.

[22] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Compt. Vis.*, vol. 42, pp. 145–175, May 2004.

[23] S. Khan, F. Sohel, and R. Togneri, "Geometry driven semantic labeling of indoor scenes," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2014, pp. 679–694.

[24] A. Vailaya, A. Jain, and H. Zhang, "On image classification: City vs. landscape," in *Proc. IEEE Workshop Content Based Access Image Video Libraries*, Jun. 1998, pp. 3–8.

[25] N. Serrano, A. E. Savakis, and J. Luo, "Improved scene classification using efficient low-level features and semantic cues," *Pattern Recognit.*, vol. 37, no. 9, pp. 1773–1784, Sep. 2004.

[26] L. Xie, J. Wang, B. Guo, B. Zhang, and Q. Tian, "Orientational pyramid matching for recognizing indoor scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, Jun. 2014, pp. 3734–3741.

[27] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the Fisher vector: Theory and practice," *Int. J. Comput. Vis.*, vol. 105, no. 3, pp. 222–245, 2013.

[28] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 652–660.

[29] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph CNN for learning on point clouds," *ACM Trans. Graph.*, vol. 38, no. 5, pp. 1–12, 2019.

[30] H. Thomas, C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6410–6419.

[31] Q. Hu et al., "RandLA-net: Efficient semantic segmentation of large-scale point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11108–11117.

[32] G. Qian et al., "PointNeXt: Revisiting PointNet++ with improved training and scaling strategies," 2022, *arXiv:2206.04670*.

[33] Y.-Q. Yang et al., "Swin3D: A pretrained transformer backbone for 3D indoor scene understanding," 2023, *arXiv:2304.06906*.

[34] X. Wang, S. Liu, X. Shen, C. Shen, and J. Jia, "Associatively segmenting instances and semantics in point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4091–4100.

[35] L. Zhao and W. Tao, "JSNet: Joint instance and semantic segmentation of 3D point clouds," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 12951–12958.

[36] L. Jiang, H. Zhao, S. Shi, S. Liu, C. W. Fu, and J. Jia, "PointGroup: Dual-set point grouping for 3D instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 4866–4875.

[37] T. Vu, K. Kim, T. M. Luu, T. Nguyen, and C. D. Yoo, "SoftGroup for 3D instance segmentation on point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 2698–2707.

[38] J. Sun, C. Qing, J. Tan, and X. Xu, "Superpoint transformer for 3D scene instance segmentation," 2022, *arXiv:2211.15766*.

[39] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe, "Mask3D: Mask transformer for 3D semantic instance segmentation," 2022, *arXiv:2210.03105*.

[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[41] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent.*, Sep. 2016, pp. 1–14.

[42] X. Wang, M. Zhu, D. Bo, P. Cui, C. Shi, and J. Pei, "AM-GCN: Adaptive multi-channel graph convolutional networks," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2020, pp. 1243–1253.

[43] M. Schlichtkrull et al., "Modeling relational data with graph convolutional networks," in *Proc. Eur. Semantic Web Conf. (ESWC)*, Mar. 2017, pp. 593–607.

[44] G. Bahl, M. Bahri, and F. Lafarge, "Road extraction from overhead images with graph neural networks," 2021, *arXiv:2112.05215*.

[45] H. Dai, B. Dai, and L. Song, "Discriminative embeddings of latent variable models for structured data," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Mar. 2016, pp. 2702–2711.

[46] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" 2018, *arXiv:1810.00826*.

[47] Z. Ying et al., "Hierarchical graph representation learning with differentiable pooling," in *Proc. NIPS*, Jun. 2018, pp. 4800–4810.

[48] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*.

[49] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. NIPS*, Jun. 2016, pp. 3837–3845.

[50] C. Ying et al., "Do transformers really perform bad for graph representation?" in *Proc. NIPS*, 2021, pp. 28877–28888.

[51] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. NIPS*, 2017, pp. 1024–1034.

[52] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," 2017, *arXiv:1710.10903*.

[53] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Apr. 2018, pp. 1–8.

[54] J. Lee, I. Lee, and J. Kang, "Self-attention graph pooling," in *Proc. Int. Conf. Mach. Learn. (PMLR)*, Apr. 2019, pp. 3734–3743.

[55] H. Gao, Y. Chen, and S. Ji, "Learning graph pooling and hybrid convolutional operations for text representations," in *Proc. World Wide Web Conf.*, May 2019, pp. 2743–2749.

[56] J. Huang, Z. Li, N. Li, S. Liu, and G. Li, "AttPool: Towards hierarchical feature representation in graph convolutional networks via attention mechanism," in *Proc. IEEE/CVF Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6479–6488.

[57] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2432–2443.

[58] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. NIPS*, 2019, pp. 8026–8037.

[59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, Dec. 2014, pp. 1–15.

[60] R. Ambrus, S. Claici, and A. Wendt, "Automatic room segmentation from unstructured 3-D data of indoor environments," *IEEE Robot. Autom. Lett.*, vol. 2, no. 2, pp. 749–756, Apr. 2017.

[61] P. Hübner, M. Weinmann, S. Wursthorn, and S. Hinz, "Automatic voxel-based 3D indoor reconstruction and room partitioning from triangle meshes," *ISPRS J. Photogramm. Remote Sens.*, vol. 181, pp. 254–278, Nov. 2021.

[62] Y. HaCohen-Kerner, D. Miller, and Y. Yigal, "The influence of preprocessing on text classification using a bag-of-words representation," *PLoS ONE*, vol. 15, no. 5, May 2020, Art. no. e0232525.

[63] V. Prakash Dwivedi and X. Bresson, "A generalization of transformer networks to graphs," 2020, *arXiv:2012.09699*.

[64] D. Chen, L. O'Bray, and K. M. Borgwardt, "Structure-aware transformer for graph representation learning," in *Proc. Int. Conf. Mach. Learn. (PMLR)*, 2022, pp. 3469–3489.

[65] L. Rampásek, M. Galkin, V. P. Dwivedi, A. T. Luu, G. Wolf, and D. Beaini, "Recipe for a general, powerful, scalable graph transformer," 2022, *arXiv:2205.12454*.

[66] A. Vaswani et al., "Attention is all you need," in *Proc. NIPS*, 2017, pp. 5998–6008.

[67] Y. Yuan, Z. Xiong, and Q. Wang, "ACM: Adaptive cross-modal graph convolutional neural networks for RGB-D scene recognition," in *Proc. AAAI Conf. Artif. Intell.*, Jul. 2019, vol. 33, no. 1, pp. 9176–9184.

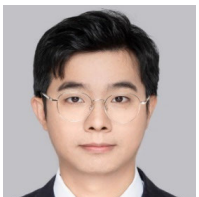
[68] R. Li, M. Tapaswi, R. Liao, J. Jia, R. Urtasun, and S. Fidler, "Situation recognition with graph neural networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 4183–4192.

Han Yue received the B.E. degree in geographical information science from the College of Geography Resource Sciences, Sichuan Normal University, Sichuan, China, in 2017. She is currently pursuing the Ph.D. degree in cartography and geographic information engineering with the College of Surveying and Geo-Informatics, Tongji University, Shanghai, China.

From October 2021 to December 2022, she was a Visiting Scholar at the Department of Earth Observation Science, ITC Faculty, University of Twente, Enschede, The Netherlands. Her research interests include indoor mapping, semantic instance segmentation, and scene classification.

Ville Lehtola received the M.Sc.(Tech) degree in physics and the Dr.Sc.(Tech.) degree in computational engineering from Aalto University, Espoo, Finland, in 2006 and 2010, respectively.

He is an Academy of Finland Postdoctoral Fellow from 2012 to 2015. Then, he was a Senior Scientist at the Finnish Geospatial Research Institute, FGI. He is currently an Assistant Professor with the Department of Earth Observation Science, ITC faculty, University of Twente, Enschede, The Netherlands. His research interests include machine learning and multisensor data processing for high-precision mobile mapping and digital twins, laser scanning and point cloud processing, and robot perception, cognition, and navigation problems.



Hangbin Wu received the Ph.D. degree in geodetics from Tongji University, Shanghai, China, in 2010.

From March to September 2009, he was a Research Assistant at the Department of Land Surveying and Geo-Informatics, Faculty of Construction and Environment, Hong Kong Polytechnic University, Hong Kong, China. He is currently an Associate Professor with the College of Surveying and Geo-Informatics, Tongji University. His research interests include the collection and processing of point clouds, high-definition maps for autonomous driving, and data mining from BIG navigation data, where he has contributed more than 50 publications.

George Vosselman received the M.Sc. degree in geodetic engineering from Delft University of Technology, Delft, The Netherlands, in 1986, and the Ph.D. degree from Rheinische Friedrich Wilhelms University of Bonn, Bonn, Germany, in 1991.

He worked as a Researcher at the Institute of Photogrammetry, Stuttgart University, Stuttgart, Germany, in 1992. After a year as a Visiting Scientist at the University of Washington, Seattle, WA, USA, where he was appointed as a Professor of photogrammetry and remote sensing at Delft University of Technology, in 1993. In 2004, he joined ITC, University of Twente, Enschede, The Netherlands, as a Professor of geo-information extraction and remote sensing. The research of his chair group focuses on the use of advancements in sensor technology for the efficient production of large-scale geo-information. He has published over 280 journals and conference papers on photogrammetry and laser scanning.

Jincheng Li received the B.E. degree in geographical information science from the College of Geography Resource Sciences, Sichuan Normal University, Sichuan, China, in 2019. He is currently pursuing the master's degree with Key Laboratory of 3-D Information Acquisition and Tourism, Capital Normal University, Beijing, China.

His research interests include laser scanning, point cloud processing, and semantic segmentation of large-scale tunnel point clouds.

Chun Liu received the Ph.D. degree in geodesy and surveying from Tongji University, Shanghai, China, in 2001.

From 2001 to 2003, he was a Visiting Scholar at the Development of Infrastructure for Cyber Hong Kong, The Hong Kong Polytechnic University, Hong Kong, China. From 2007 to 2008, he was a Senior Visiting Scholar at the Laboratory of Mapping and GIS, Ohio State University, Columbus, OH, USA. In 2011, he was a Senior Visiting Scholar at the Institute of Spatial Geodesy and Engineering Survey, University of Nottingham, Nottingham, U.K. He is currently a Full Professor of Cartography and Geography Information Engineering with the College of Surveying and Geo-Informatics, Tongji University. His research interests include multisource point cloud-coupled semantic cognition and spatial-temporal smart sensing.