Leveraging Commonsense for Object Localisation in Partial Scenes

Francesco Giuliari, *Student Member, IEEE*, Geri Skenderi, Marco Cristani, *Member, IEEE*, Alessio Del Bue, *Member, IEEE*, and Yiming Wang

***Abstract*—We propose an end-to-end solution to address the problem of object localisation in partial scenes, where we aim to estimate the position of an object in an unknown area given only a partial 3D scan of the scene. We propose a novel scene representation to facilitate the geometric reasoning, Directed Spatial Commonsense Graph (D-SCG), a spatial scene graph that is enriched with additional concept nodes from a commonsense knowledge base. Specifically, the nodes of D-SCG represent the scene objects and the edges are their relative positions. Each object node is then connected via different commonsense relationships to a set of concept nodes. With the proposed graph-based scene representation, we estimate the unknown position of the target object using a Graph Neural Network that implements a sparse attentional message passing mechanism. The network first predicts the relative positions between the target object and each visible object by learning a rich representation of the objects via aggregating both the object nodes and the concept nodes in D-SCG. These relative positions then are merged to obtain the final position. We evaluate our method using Partial ScanNet, improving the state-of-the-art by 5.9% in terms of the localisation accuracy at a 8x faster training speed.**

***Index Terms*—Vision and scene understanding, scene analysis, computer vision, machine learning.**

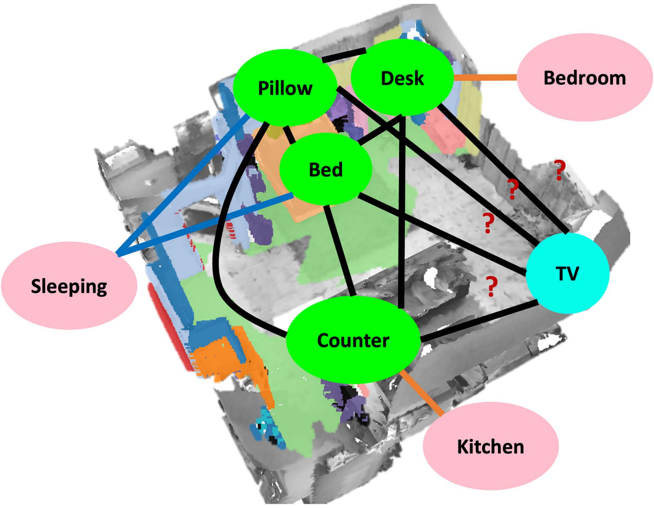


Fig. 1. Given a set of objects (*green* nodes) in a partially known scene that is semantically segmented, we aim to estimate the position of a target object (the *cyan* node) in the unexplored (grey) area. We address this localisation problem with a novel scene graph representation dubbed **D-SCG**, that contains both the spatial knowledge extracted from the reconstructed scene, i.e., the proximity (*black* edges), and the commonsense knowledge represented by a set of relevant concepts (*pink* nodes) connected by relationships, e.g., *UsedFor* (*blue* edges) and *AtLocation* (*orange* edges).

I. Introduction

LOCALISING an unobserved object given only a partial observation of a scene, as shown in Fig. 1, is a fundamental task in many automation applications such as object search with embodied agents [1], layout generation for interior layout design [2], and for assisting visually impaired people in finding everyday items. Humans can perform such a task with ease, using the past experience and the fact that there exists some commonsense in terms of object arrangement patterns within specific scenarios. For example, when we arrange objects in a house, we often place the television in front of a sofa in the living room, and put the nightstand beside the bed in the bedroom.

In this work, we present a novel solution for the localisation of objects in partially observed 3D scenes. Our method is able to infer the position of an object in the unobserved part of the room by leveraging the commonsense knowledge together with the geometric arrangement of objects in the visible part. We demonstrate that the *commonsense* knowledge that indicates *how objects are used*, e.g., a chair is used for sitting, and *where they are typically located*, e.g., a chair is often located in a kitchen, can be exploited to improve the accuracy of the localisation.

As shown in Fig. 1, we propose to model the objects' arrangement and commonsense information as a heterogeneous graph called Directed Spatial Commonsense Graph (**D-SCG**).

Manuscript received 17 June 2022; revised 31 January 2023; accepted 26 April 2023. Date of publication 3 May 2023; date of current version 5 September 2023. This work was supported in part by the European Union's Horizon 2020 research and innovation programme "MEMEX" under Grant 870743, and in part by the Italian Ministry of Education, Universities and Research (MIUR) through PRIN 2017 - Project under Grant 20172BH297: I-MALL and "Dipartimenti di Eccellenza 2018-2022". Recommended for acceptance by V. Lepetit. *(Corresponding author: Francesco Giuliari.)*

Francesco Giuliari is with the Istituto Italiano di Tecnologia (IIT), 16163 Genoa, Italy, and also with the University of Genoa (UniGe), 16124 Genova, Italy (e-mail: francesco.giuliari@iit.it).

Geri Skenderi is with the University of Verona (UniVr), 37129 Verona, Italy (e-mail: geri.skenderi@univr.it).

Marco Cristani is with the University of Verona (UniVr), 37129 Verona, Italy, and also with the Istituto Italiano di Tecnologia (IIT), 16163 Genoa, Italy (e-mail: marco.cristani@univr.it).

Alessio Del Bue is with the Istituto Italiano di Tecnologia (IIT), 16163 Genoa, Italy (e-mail: alessio.delbue@iit.it).

Yiming Wang is with the Fondazione Bruno Kessler (FBK), 38123 Povo, Italy, and also with the Istituto Italiano di Tecnologia (IIT), 16163 Genoa, Italy (e-mail: ywang@fbk.eu).

Digital Object Identifier 10.1109/TPAMI.2023.3272523

First, the nodes representing the known objects in the partially observed scene construct a Directed Spatial Graph (**D-SG**), which is fully connected. The edges between the nodes are called proximity edges, representing the *relative position* between a pair of objects. Then, the **D-SG** is further expanded into the **D-SCG** by adding and connecting nodes that represent concepts through relevant commonsense relationships extracted from ConceptNet [3]. The object to locate, i.e., the target, is a node in the graph, connected to the other known object nodes with proximity edges, where the respective relative positions are treated as unknowns. Our proposed solution for object localisation exploits a Graph Neural Network (GNN) that can efficiently learn the representations of the nodes and edges of **D-SCG**. Our network predicts the relative positions between the target object node and each known object node. It then converts the relative positions into absolute ones which agree on a single final position of the target object.

As an extension of [4], our new **D-SCG** represents the proximity edges with directional relative positions. Instead, the method in [4] localises objects with Spatial Commonsense Graph (SCG-OL) that connects object nodes with undirected proximity edges defined by the relative object distances. This difference in the graph formulation allows us to regress and estimate the target position in an *end-to-end* trainable manner. This is not possible with SCG-OL [4] due to the requirement of a non-differentiable multilateration procedure for localising the target object. The novel scene graph formulation also leads to a simpler loss calculation and training procedure which benefits the encoding of both the geometrical information regarding the arrangement of the objects in the observed part of the room and commonsense attributes that define what they are *commonly* used for or where they are *commonly* located, resulting in a better target localisation both on the 2D floor map and in the 3D scene. Moreover, we propose to employ a new attention module in the GNN that is adapted from the Rectified Linear Attention (ReLA) [5] for its high expressive power that encourages the sparsity of attention weights, while being stable and efficient in terms of training. With extensive experiments, we demonstrate that our new method achieves an increase of 5.9% in terms of Localisation Success Rate (LSR), compared to the previous state-of-the-art method SCG-OL, with a 8x speed-up in both training and inference. Furthermore, the proposed solution is able to generalise to the 3D domain, reaching 25% in terms of LSR compared to the 5% of SCG-OL.

The main contributions of this paper are summarised below:

- We introduce the *Directed Spatial Commonsense Graph* (**D-SCG**), an heterogeneous scene graph representation that integrates both the spatial information of the partially observed scene and the commonsense knowledge that is relevant to the observed objects. **D-SCG** defines its proximity edges with the directional relative positions, different from the formulation in [4] that represents the proximity edges as distances. This allows for the design of an efficient end-to-end learnable localisation network, leading to a better localisation performance in both 2D and 3D.
- We propose ***D-SCG** Object Localiser*, a GNN-based solution that uses the **D-SCG** for the localisation of objects in the unobserved part of the scene. We utilise a new sparse attention module that is adapted from [5] in our GNN, contributing to a higher object localisation success rate and training efficiency.
- We present an extensive evaluation of our solution, as well as an in-depth analysis of the internal working of the proposed model, to explain the utility of commonsense reasoning.

II. Related Work

We cover related works regarding the use of graphs for modelling scenes and performing inference, and the use of commonsense reasoning in neural networks.

Scene Graph Modelling and Inference. Scene graphs allow high-level description of a scene by its content. They were initially introduced to model the objects in an image and their relations. In most applications, the nodes in the scene graph indicate the objects in the image, and the edges define the relationships between these objects, which can be spatial [6] or semantical [7]. Scene graphs are useful in many applications, such as image retrieval [7], [8], image captioning [9], [10], [11] and visual question answering [12], [13], where using an abstract representation of the scene is better than directly working with pixels. Scene graphs have also been shown to be useful in improving classical tasks like object detection by allowing reasoning on contextual cues from neighbouring objects [6].

Recently, their usage has been extended to 3D, providing an efficient solution for 3D scene description. The scene graphs for 3D applications vary from simple structures, where the nodes define the objects in the scene, and the edges define the spatial relationship between them [14], [15], [16], to more complex, hierarchical structures. Armeni et al. [17] propose a hierarchical scene graph for large-scale environments that can encode information at different "levels" with each level providing a more abstract representation. The first level records the data retrieved from the individual cameras, such as images and camera positions. The second level provides information regarding the objects in the environment, the third is about the rooms, and finally, the last one is about the buildings. Such representation is ideal for large environments, but it is needlessly complex for most applications where the environment is composed of a few rooms, where more straightforward representations are typically used. 3D scene graphs are commonly used for layout completion, scene synthesis and robot navigation [18], [19], [20]. Zhou et al. [18] uses a GNN in combination with a 3D scene graph to enrich indoor rooms with new objects that match their surroundings. In [19], the layout of the scene is encoded using a relation graph with objects as nodes and spatial/semantic relationships between objects as edges. The relation graph is then used to train a generative model that produces novel relation graphs, thus new layouts. For robot navigation, the scene graph is used to encode the environment efficiently. In [20] the scene graph is used to encode the scene's geometry, topology, and semantic information. The scene graph is then mapped to the robot's control space for navigation via a learned policy.

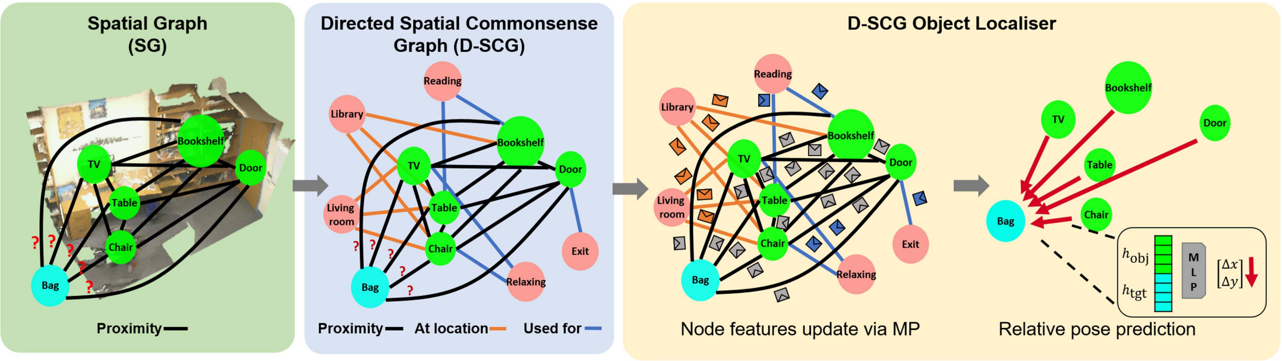


Fig. 2. General overview of our proposed approach. First, we construct the Directed Spatial Commonsense Graph (**D-SCG**) from the known scene by enriching the directed scene graph with concept nodes and relationships, resulting in edges of three types: *UsedFor* (*Blue* edges), *AtLocation* (*Orange* edges) and Proximity (edges). The **D-SCG** is then fed into our **D-SCG Object Localiser** that first performs message passing with attention to update the node features taking into consideration the heterogeneous edges, then concatenates the node features of the target node and one of the scene object nodes (at a time) and passes it through an MLP to predict the relative position. The final position is given by aggregating all the predicted relative positions via mean pooling.

Commonsense Knowledge in Neural Networks. Commonsense reasoning refers to the high-level reasoning that humans employ when solving tasks. In particular, it is our ability to use prior information gained in our lifetime and use it for a new task. While modelling human-level commonsense is something that we are still far away from achieving, much work has been done in this direction in recent years. A fundamental requirement is to have a way to provide “known prior information”. This is typically achieved using knowledge bases that are considered to contain some kind of axiomatic truths regarding our world. Examples of such knowledge bases are WikiData [21] and ConceptNet [3].

In the field of Natural Language Processing (NLP), the work presented in [22] makes use of ConceptNet to create richer contextualised sentence embeddings with the BERT architecture [23]. In [24], the authors utilise the knowledge graph Freebase to enrich textual representations for a question answering system. In computer vision, a few works [25], [26] have exploited an external knowledge base for Visual Question Answering (VQA) as it helps the network to reason beyond the image contents. In the scene graph generation task, the ConceptNet [3] knowledge graph has also been exploited to refine object and phrase features to improve the generalisation of the model [27]. In this work, we incorporate commonsense information from ConceptNet into the spatial scene graph to improve the object localisation performance when only a partial scene is observed.

III. Directed Spatial Commonsense Graph

Our scene representation has the objective to embed commonsense knowledge into a geometric scene graph extracted from a partial 3D scan of an area. As shown in Fig. 2, we construct the **D-SCG** with nodes that are: *i)* object nodes that include all the observed objects in the partial scene and any unseen target object to be localised; *ii)* concept nodes that are retrieved from ConceptNet [3].

Each **D-SCG** is constructed on top of a directed Spatial Graph (**D-SG**), a fully directed graph with all object nodes. Each object node is further connected to a set of concept nodes via some semantic relationships available in the knowledge base. This renders the edges of **D-SCG** heterogeneous, separating the spatial interactions from the “commonsense”. In practice, the edges of our proposed graph structure are of three types:

- *Proximity*, represented by the relative position vector, indicating both the distance and direction, between *all* object nodes given the partial 3D scan. This is different from our previous work [4] where *Proximity* is represented by the relative distance between *all* the object nodes of the partial scene;
- *AtLocation*, retrieved from ConceptNet, indicating in what environment the objects are often located in;
- *UsedFor*, retrieved from ConceptNet, describing common use-cases of the objects.

The proximity edges connect all the objects nodes of the **D-SCG** in a directed and complete manner, while the semantic *AtLocation* and *UsedFor* edges connect each object node with its related concept nodes that are queried from ConceptNet (e.g., *bed AtLocation apartment* or *bed UsedFor resting*). The two semantic edge types provide useful hints on how objects can be clustered in the physical space, thus benefitting the position inference of indoor objects.

We formulate **D-SCG** as a directed graph that is composed by a set of nodes $\mathcal{H} = \{h_i | \, i \in (0, N]\}$, where $N = N_o + N_c$ is the total number of nodes. N_o is the number of the object nodes and N_c the number of the concept nodes, where each node is represented by a feature vector $h_i \in \mathbb{R}^{300}$ from ConceptNet [28]. The edges are defined by the set $\mathcal{E} = \{e_{i,j} | \, i \in (0, N], j \in \mathcal{N}_i\}$, where $e_{i,j}$ is the edge between node i and node j and $\mathcal{N}_i$ is set of neighbouring nodes of i.

We represent each edge with a 6-dimensional feature vector, i.e., $e_{i,j} \in \mathbb{R}^6$, whose first three elements indicate the edge type in a one-hot manner, the fourth element indicates whether a proximity relation involves the target node, while the last two elements indicate the relative position $d_{i,j} = [\Delta x_{i,j}, \Delta y_{i,j}]$ between node i and node j, in Cartesian coordinates such that $\Delta x_{i,j} = x_j - x_i$ and $\Delta y_{i,j} = y_j - y_i$. This definition is different from the **SCG** in [4], where edges were represented by 4-dimensional vectors that represented the one-hot encoded edge class and only the distance between the objects that were connected by the edge. In our new graph formulation, we are able to achieve a more detailed spatial reasoning and to train in an end-to-end manner, which contributes to the performance improvement in both object localisation and computational efficiency as shown in Section V-B. As the relative positions are only measurable among object nodes in the observed part of the 3D scan, we initialise the relative positions to [0,0] when the edges are *AtLocation*, *UsedFor*, or *Proximity* edges involving the unknown target object node.

Note that we focus on localising the target object on the XY plane since the target's positions vary little along the Z axis, as indicated in the benchmark's statistics, shown in [4]. Nevertheless, our method can easily be extend to perform predictions in 3D space by predicting the relative 3D positions $d_{i,j} = [\Delta x_{i,j}, \Delta y_{i,j}, \Delta z_{i,j}]$ between nodes i and j, in Cartesian coordinates. We provide experimental results on 3D localisation in Section V-C.

IV. End-to-End D-SCG Object Localiser

We propose an end-to-end solution to address the task of localising the arbitrary unobserved target object using the **D-SCG**. The model first predicts the relative positions of the unseen target object w.r.t. the objects in the partially known scene. Then the relative positions are converted in absolute coordinates and mean pooling is applied to estimate the final position. This approach is fully differentiable and requires no additional localisation module based on circular triangulation to predict the position of the target object as in [4], thus improving the network's efficiency.

A. Model

To predict the relative position of the unseen target node w.r.t. the visible scene objects, we make use of a stacked GNN architecture. Our proposed GNN replaces the attention mechanism in Graph Transformer [29] with an sparse attentional message passing mechanism that is based on ReLA [5]. We further add a ScaleNorm layer on top of ReLA and utilise different normalisation layers to stabilise the training of the new module on our **D-SCG**. The node embeddings are updated iteratively by utilising the heterogeneous information of the edge type, to allow effective fusion between the commonsense knowledge and the metric measurements. We highlight the main differences between the attention mechanism in [29] and ours in Fig. 3.

The input to the network is a set of node features $\mathcal{H}$ and the output is a new set of node features $\mathcal{H}' = \{h'_i | \, i \in (0, N]\}$, with $h'_i \in \mathbb{R}^{300}$. Each node i in the graph is updated by aggregating the features of its neighbouring nodes $\mathcal{N}_i$ via four rounds of message passing. The resulting h'_i forms a *contextual* representation of its neighbourhood.

At each round of the message passing, we learn an attention coefficient $\alpha_{i,j}$ between each pair of connected nodes using a graph-based and rectified version of the scaled dot-product attention mechanism, conditioned on the node and edge features. Our GNN can learn sparse and (positively) unbounded attention weights due to the usage of the activation function ReLU, as proposed for the vanilla Transformer model by [5], thus allowing for the understanding of arbitrary relationships between the different node types.

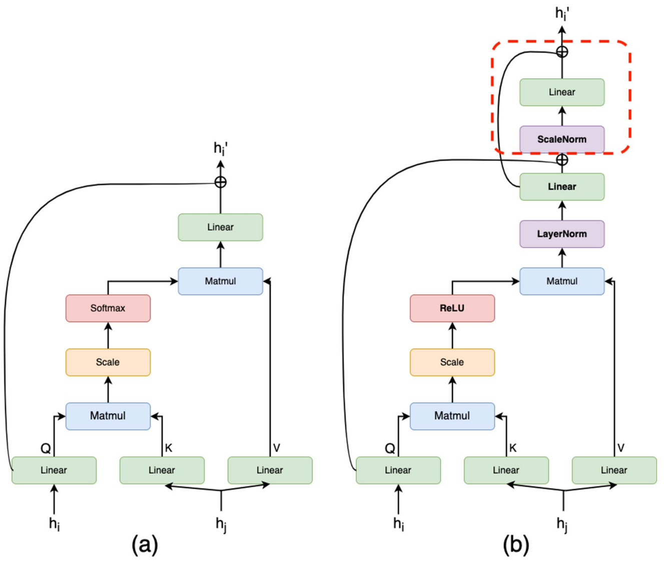


Fig. 3. Overview of the differences between the attention mechanism of [29] (a), used in **SCG** [4], and the one employed in this paper that is based on ReLA [5] with an added ScaleNorm and reprojection (highlighted in the red dashed box)(b). The new attention mechanism contains more parameters, thus producing more expressive representations, and learns sparse weights with reduced training and inference time thanks to the ReLU activation function. Moreover, we utilise two different normalisation layers to stabilise the network's training.

The network starts by performing an affine transformation of the relevant node and edge features to calculate the corresponding query, key, value and edge vector that will be used to compute the attention weights:

$$q_i = W_q h_i + b_q, \tag{1}$$

$$k_j = W_k h_j + b_k, \tag{2}$$

$$v_j = W_v h_j + b_v, \tag{3}$$

$$e_{ij} = W_e e_{ij} + b_e, \tag{4}$$

where W and b represent respectively the learnable weight matrices and bias vectors for each transformation.

The network then calculates the attention weight $\alpha_{i,j}$ between two nodes i and j as:

$$\alpha_{i,j} = ReLU\left(\frac{\langle q_i, k_j + e_{ij}\rangle}{\sqrt{d}}\right) \tag{5}$$

where $\sqrt{d}$ is a scaling term equal to the square root of the dimension of the projected features k_j. As seen in (5), the use of the softmax is dropped, since it involves aggregating the scores for all the edges connected to each node, which implies many operations for large graphs and slows down the training. Additionally, the use of ReLU allows for sparsity in the attention weight matrix, which helps analyse how the network prioritises the exchange of information. As the attention weights that are calculated using ReLU are not limited to the range (0,1), we use

Layer Normalisation [30] when calculating the updated node features h'_i, followed by a *gated residual connection* that prevents the node features from converging into indistinguishable features [29]:

$$h'_i = LayerNorm(h_i + \langle \alpha_{i,j}, v_j + e_{ij} \rangle) \quad (6)$$

$$\beta_i = Sigmoid(W_g[h'_i; W_r h_i + b_r; h'_i - (W_r h_i - b_r)]) \quad (7)$$

$$h'_i = (1 - \beta_i)h'_i + \beta_i(W_r h_i + b_r), \quad (8)$$

where [;] represents the concatenation operation, $LayerNorm$ is the Layer Normalisation, β_i is a learnable parameter that guides the gated residual connection, and W and b are the learnable weight matrices and bias vector for the respective linear transformations.

Different from [5], we then re-normalise and re-project these features in a similar fashion to the original Transformer model, a practice that has been empirically shown to stabilise and improve the training of self-attentive neural networks [31].

$$h'_i = ScaleNorm(W_o h'_i + b_o), \quad (9)$$

This step further increases the number of learnable parameters of our GNN, allowing for better scaling and more expressive representations, while not sacrificing efficiency thanks to the sparse attention mechanism (described previously) and the Scale Normalisation introduced in [31].

We combine all these operations in a module, and use it for a total of four message passing rounds. Finally, we obtain the set of final node embeddings $\mathcal{H}^* = \{h^*_i | \, i \in (0, N]\}$, with $h^*_i = [h_i; h'_i]$. In this way, the final representation of each node contains both the original object embedding and the aggregated embedding of its context in the scene. Finally, we concatenate the features of the two nodes $h^*_{i,t} = [h^*_i; h^*_t]$, and predict the relative position $\hat{d}_{i,t}$ between the target object node t and the observed object node i via linear projection. To obtain the final position, we first convert the relative positions $\hat{d}_{i,t}$ in absolute coordinates by summing to them the positions $p_i = [x_i, y_i]$ of the observed object nodes and then take the mean of the absolute positions as our predicted position $\hat{p}_t$.

B. Loss

We train our network with a strategy which considers that multiple instances of the searched object can exist in the unobserved part of the scene. Therefore, only the instance closest to the prediction is accounted when calculating the loss. By doing this, the network learns to correctly predict a specific position instead of a point that minimises the distance w.r.t. all the instances. For the loss, we minimise the squared L2 distance between the predicted position $\hat{p}_t$ and the ground-truth position of the target position p_t as follows:

$$\mathcal{L}_2(\hat{p}_t, p_t) = \|\hat{p}_t - p_t\|^2_2. \quad (10)$$

V. Experiments

We evaluate our proposed method on a dataset of partially reconstructed indoor scenes. In the following sections, we first give some relevant details on the partial scene dataset in Section V-A. Then we present comparisons of our proposed method against the state-of-the-art methods, accompanied by the implementation details, evaluation metrics and discussions in Section V-B. Finally, we show different ablation studies in Section V-C to prove our main design choices and to demonstrate some interesting aspects of our method, including how the proposed attention evolves over message passing and the extension towards 3D object localisation.

A. Dataset

Our training and evaluation is based on the partial 3D scenes dataset [4]. The dataset is built using data from ScanNet [32] which contains RGB-D sequences taken at a regular frequency with a RGB-D camera, providing the camera position corresponding to each captured image, as well as the point-level annotations, i.e., class and instance id, for the complete Point Cloud Data (PCD) of each reconstructed scene.

As the original acquisition frequency in ScanNet is very high (30 Hz), the partial scene dataset only uses a subset provided in the ScanNet benchmark[1] with a frequency of about $1/100$ of the initial one. Each full RGB-D sequence of each scene is divided into smaller sub-sequences to reconstruct the partial scenes, with varying length to reflect different levels of completeness of the reconstructed scenes (see Fig. 4 for an example). For each sub-sequence, the RGB-D information is integrated with the camera intrinsic and extrinsic parameters to reconstruct the PCD at the resolution of 5 cm using Open3D [33]. The annotation for each point in the partial PCD is obtained by looking for the corresponding closest point in the complete PCD scene provided by ScanNet.

We extract the corresponding **D-SG**, i.e., the graph with only proximity edges, for each partially reconstructed scene and its object nodes. The nodes of the graph contain the object information: its *position*, defined as the centre of the bounding box containing the object and the *object class*. The proximity edge connects two object nodes and contains the relative position of the second object with respective to the first. We consider the position of each scene object as a 2D point (x, y) on the ground plane as most objects in the indoor scenes of ScanNet are located at a similar elevation. Each node is marked as *observed* if it represents an object in the partially known scene; or as *unseen* if it represents the target object in the unknown part of the scene.

On top of **D-SG**, each **D-SCG** is constructed by adding two semantic relationships *AtLocation* and *UsedFor*, as well as the concept nodes that are linked to the scene nodes by these relationships. The concepts are extracted from ConceptNet by querying each scene object using the two semantic relationships. The query returns a set of related concepts together with their corresponding weight w, which indicates how "safe and credible" each related concept is to the query. We include a concept to the **D-SCG** only when it has a weight $w > 1$. Fig. 6 shows an example of a scene and the extracted **D-SCG**. Fig. 5 shows the average number of nodes linked by different types of edges in the **D-SCG**. On average, each **D-SCG** contains about

[1] http://kaldir.vc.in.tum.de/scannet_benchmark

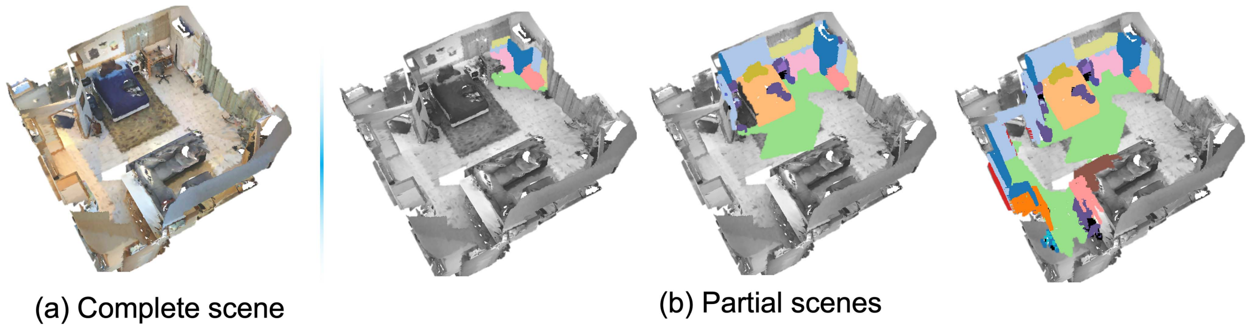


Fig. 4. The dataset with (a) the complete scene from the ScanNet [32] dataset, and (b) the extracted partial scenes [4]. The observed part of the scene is coloured based on the object semantics, while the unexplored part of scene is coloured in grey.

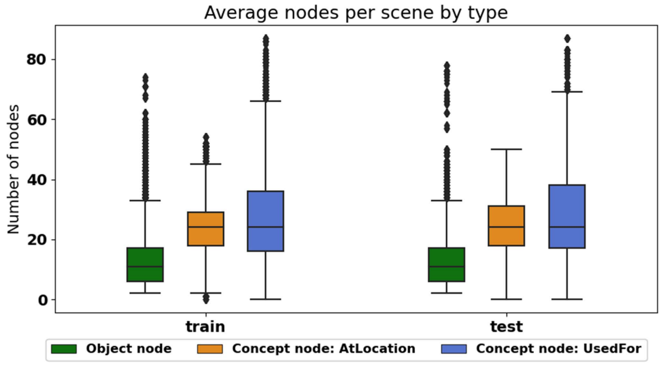


Fig. 5. Average number of different types of nodes in the **D-SCG** for both train and test splits of the dataset. The outliers in the boxplots are introduced by uncommon room types with a large amount of objects, e.g., libraries with several books.

5 times more the concept nodes than the object nodes in the **D-SG**, demonstrating that a rich commonsense knowledge is included in **D-SCG**.

Finally, we follow the same training/validation/test split as in [4], using 19461 partial scenes for training and validation, and 5435 partial scenes for testing, with each partial scene having its corresponding **D-SG** and **D-SCG**.

B. Experimental Comparisons

We validate **D-SCG-OL** by comparing its performance against a set of baselines, a method for layout prediction adapted for the localisation task [34] and the state-of-the-art approach SCG-OL for object localisation [4]. The baselines and SCG-OL follow the two-staged pipeline, where they first predict the pairwise distances and then estimate the position by using a localisation module, which minimises these pairwise distances (circular intersection). We summarise all the approaches implemented for evaluation below.

- *Statistics-based baselines* use the statistics of the training set, i.e., the *mean*, *mode*, and *median* values of the pairwise distances between the target object and the scene objects, as the predicted distance.
- *MLP* learns to predict pairwise distances between the target object and every other observed object in the scene without considering the spatial or semantic context. The input to this model is a pair of the target object and the observed object with each object represented by a one-hot vector indicating the class, passed to an MLP that predicts pairwise distances.
- *MLP with Commonsense* learns to predict the pairwise distance between the target object and every other observed object in the scene without considering the spatial context. We first use GCN to propagate the conceptnet information to object nodes, then the features are passed to a MLP that predicts pairwise distances.
- *LayoutTransformer* [34] uses the transformer's self-attention to generate the 2D/3D layout in an auto-regressive manner. We describe the observed objects as a sequence of elements as in [34], where each element contains the object class and the position (x, y). We then feed the class of the target object to generate its corresponding position (x, y). For a fair comparison, we retrain the model on our training set.
- **SCG-OL** [4] exploits **SCG** that contains the proximity edges that describe the relative distances. The localisation method is a two-stage approach which first predicts the pairwise distances via a stacked GNN, and then passes these to a Localisation module to obtain the final position.
- ***D-SCG-OL*** *w\o Commonsense* is a variant of our approach to test the capability of the method when it is used without commonsense knowledge. The input is the **D-SG**, which is composed only by the object nodes and proximity edges. The initial node features are not pre-trained word embeddings, but are learned during training via an embedding layer.
- Graphormer [35] is a recent work that adapted Transformer models [36] to graph learning. It uses different embedding strategies to add inductive bias related to the graph structure. It then propagates features between nodes up to k-hop distances using the attention mechanism. We adapted the method so that the node features are updated using the Graphormer network, then the target node features are used to regress the position of the searched object.
- **D-SCG-OL** is our proposed method, described in Section IV, along with a variant that is trained with learnable node embeddings, instead of initialising each node with the commonsense embedding coming from ConceptNet's Numberbatch.

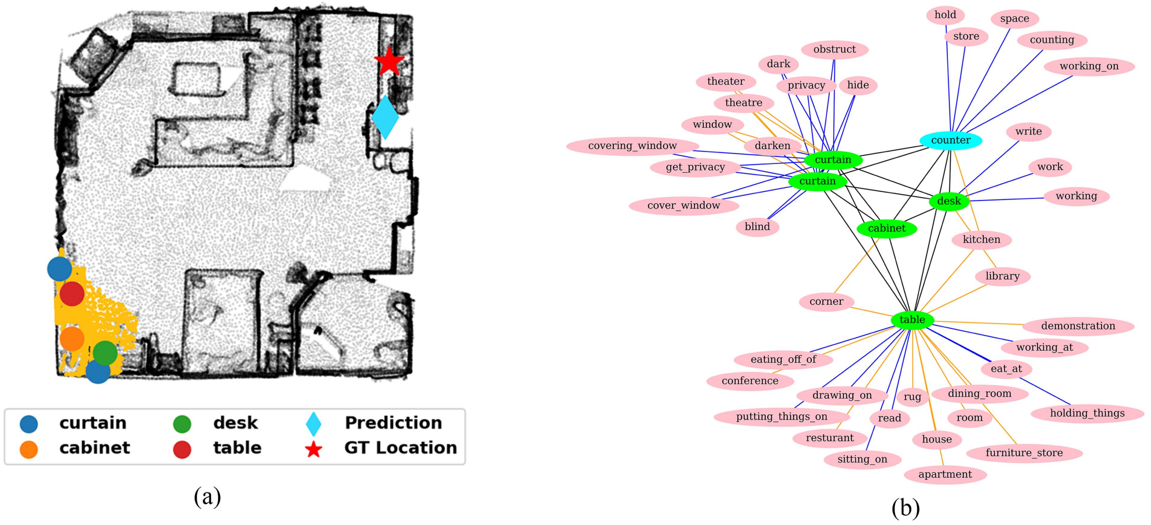


Fig. 6. Example of a Partial Scene and its generated **D-SCG**. The target object is represented by the cyan node, the scene objects are the green nodes, and the concept nodes have a pink background. The colour of the edge distinguishes the relationship type: orange ones are *AtLocation* edges, blue ones are *UsedFor* edges, and *black ones* are *Proximity* edges. (a) Scene. (b) Directed spatial commonsense graph.

We evaluate the different methods for the localisation on the 2D floor plane, and report additional results for 3D localisation in the ablation studies.

Evaluation Measures. We evaluate the performance in terms of the successful target object localisation and the relative pairwise distances, as also proposed in [4].

- *Localisation Success Rate (LSR)* quantifies the localisation performance. LSR is defined as the ratio of the number of successful localisations over the number of tests. A localisation is considered successful if the predicted position of the target object is close to a target instance within a predefined threshold. Unless stated differently, the distance threshold is set to 1 m. We consider LSR as the *main* evaluation measure for our task.
- *mean Successful Localisation Error (mSLE)* quantifies the localisation error among successful cases. mSLE is the Mean Absolute Error (MAE) between the predicted target position and the ground-truth position among all successful tests.
- Finally, *mean Predicted Proximity Error (mPPE)* quantifies the performance of the methods that rely on pairwise relative distance prediction, as described above. mPPE is the mean absolute error between the predicted distances and the ground-truth pairwise distances between the target object and the objects in the partially known scene.

Implementation Details. We train our network using the Adafactor optimiser [37] for 200 epochs. We use a total of 4 message passing layers, a number that is carefully chosen (see details in Section V-C). The dimension of the first message passing projection is set to $D = 256$ and $2D$ for the remaining rounds. All attention modules use 4 attention heads. During training, we augment the dataset by applying random rotations to the scene objects to allow for better generalisation.

Discussion. Table I reports the localisation performance measures in terms of mPPE, LSR, and mSLE, of all compared methods evaluated on the dataset with partially reconstructed scenes. We can initially observe that methods which rely *only* on pairwise inputs, e.g., statistics-based approaches or MLP, lead to worse performance compared to methods that account for other objects present in the observed scene. Nevertheless, introducing semantic reasoning on top of these methods seems to improve the performances, as shown by MLP w/ Commonsense, with an improved LSR of 2% compared to the standard MLP. LayoutTransformer directly predicts the 2D position of the target object by taking as input the list of all the observed scene objects and using the target class as the last input token. LayoutTransformer can better encode the spatial context and outperforms the statistic-based and MLP baselines. **SCG-OL** that uses the **SCG** with pairwise distances is able to improve on all metrics w.r.t. the baseline methods, suggesting that a scene-graph based solution with added commonsense knowledge is a more effective

TABLE I

METHODS COMPARISON FOR OBJECT LOCALISATION IN PARTIAL SCENES. MPPE: MEAN PREDICTED PROXIMITY ERROR. MSLE: MEAN SUCCESSFUL LOCALISATION ERROR. LSR: LOCALISATION SUCCESS RATE. SG: SPATIAL GRAPH. **SCG**: SPATIAL COMMONSENSE GRAPH. **D-SG**: DIRECTED SPATIAL GRAPH. **D-SCG**: DIRECTED SPATIAL COMMONSENSE GRAPH. THE FIRST PART OF THE TABLE FOLLOW THE 2-STAGE APPROACH WHICH FIRST PREDICTS THE PAIRWISE DISTANCES AND THE LOCALISE THE OBJECT VIA MULTILATERATION. THE LAST PART CONSISTS OF OUR METHOD AND ITS VARIANTS WHICH DIRECTLY PREDICT THE FINAL POSITION

Method	Data type	mPPE(m)↓	mSLE(m)↓	**LSR ↑**
Statistics-Mean	Pairwise	1.167	0.63	0.140
Statistics-Mode	Pairwise	1.471	0.63	0.149
Statistics-Median	Pairwise	1.205	0.64	0.164
MLP	Pairwise	1.165	0.62	0.143
MLP w/ Commonsense	Pairwise	1.090	0.64	0.163
LayoutTransformer [34]	List	-	**0.59**	0.176
SCG-OL- Learned Emb [4]	**SCG** [4]	0.974	0.61	0.234
SCG-OL [4]	**SCG** [4]	0.965	0.61	0.238
Graphormer [35]	**D-SCG**	-	0.59	0.251
D-SCG-OL w/o Commonsense	**D-SG**	-	0.59	0.265
D-SCG-OL - Learned Emb	**D-SCG**	-	0.57	0.273
D-SCG-OL	**D-SCG**	-	0.55	**0.297**

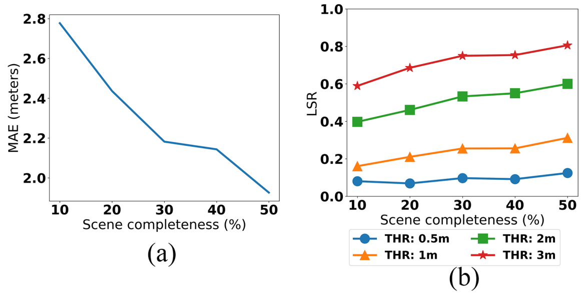


Fig. 7. Localisation performance over different levels of scene completeness. (a) The localisation error in terms of MAE between the estimated target position and the ground-truth position. (b) The LSR at different threshold levels.

way of modelling the problem. The adapted Graphormer [35] model performs the best amongst the baselines, but it fails to reach the same performances as our proposed approach. Graphormer proposes to enhance the propagation of information by aggregating the information not only from directly connected nodes but also from nodes up to a k-distance by creating new edges between them where the edge features are an inner product between all the edges along the path. While this works fine on a homogeneous graph, the effect can be limited with heterogeneous edge types, since the structural information of our proposed heterogenous graph is not as meaningful as the molecular graphs, for which Graphormer was proposed.

The different versions of our proposed method **D-SCG-OL** are able to reach the best performance. **D-SCG-OL** with learned embeddings has a 0.8% increase in the LSR performance w.r.t. the GNN working only on the **D-SG**, revealing the usefulness of the concept nodes, with a further increase of 2.4% when initialising the node embeddings of the graph by using ConceptNet's Numberbatch, showing that the commonsense information introduced from ConceptNet is useful for the localisation task.

As previously mentioned in this section, we consider the LSR as the primary evaluation metric. It is therefore useful to demonstrate and understand how the completeness (known) level of the scene impacts the localisation performance of **D-SCG-OL**. Fig. 7(a) reports the mean absolute error (MAE) between the estimated position and the ground-truth position compared to the scene completeness. Note that the MAE is calculated on all the test cases, including both the successful and failed ones. We use MAE instead of the mSLE as the mSLE is calculated only on successful cases and does not change with the completeness of the scene. As a general trend, our model can predict more accurately the position of the target object with an increasing scene completeness. Fig. 7(b) presents how the LSR varies as the scene gets more complete. In general, the LSR increases when the localisation error decreases. We report the LSR at four different threshold values, i.e., 0.5 m, 1 m, 2 m, and 3 m, where a larger threshold leads to a larger LSR value, as it might be expected.

Qualitative Results. Fig. 8 shows the qualitative results obtained using our method **D-SCG-OL**. Fig. 8(a) shows that the "sink" object class was successfully located near the counter. Similarly in Fig. 8(b), the position of the chair (target object) is correctly estimated in a position which is coherent with other instances of chair and tables in the observed part of the room. Interestingly, Fig. 8(c) presents a failure case in which the method fails to locate a window in an office setting. In this case the network successfully identify the general direction where the window should be located, but overestimated its concrete placement w.r.t to the visible objects. This error is plausible as the network does not see any objects that can help create an idea of the actual shape of the room.

TABLE II
IMPACTS OF DIFFERENT CONCEPTNET RELATIONSHIPS WITH THE PROPOSED **D-SCG-OL**. LSR: LOCALISATION SUCCESS RATE

Edge Types	Obj. linked by n semantic edges (%)			LSR ↑
	0	1	2	
Proximity	100	0	0	0.257
AtLocation, Proximity	8	92	0	0.292
UsedFor, Proximity	19	81	0	0.272
AtLocation, *UsedFor*, Proximity	8	12	80	**0.297**

Computational Efficiency. There are 20.5 M parameters in **D-SCG-OL**, which is 3.4 M more than the previous state-of-the-art method SCG-OL [4] (17.1 M). Nevertheless, our proposed **D-SCG-OL** takes 13h35 m to fully train the model for 200 epochs on a single Titan RTX, while SCG-OL requires 108h40 m, thus 8x slower. This is mostly due to the two-stage approach of SCG-OL which includes the non-differentiable localisation module and the more expensive activation function in the attention mechanism.

C. Ablation Studies

We further analyse **D-SCG-OL** to justify the usefulness of the commonsense relationships and our new attentional message passing mechanism. We also investigate the impact of increasing the number of message passing layers. To verify the applicability in 3D, we also evaluate the localisation performance of our method in comparison to the state-of-the-art methods. Lastly, we provide in-depth investigation on how the attention weights evolve over the message passing when forming the node and edge representation.

Which commonsense relationship is more important? In order to better understand the effects of using different commonsense relationships, we compare the performance of **D-SCG-OL** against four variants where the **D-SCG** contains: i) only *Proximity* edges without commonsense relationships, ii) *Proximity* edges with *AtLocation* edges, iii) *Proximity* edges with *UsedFor* edges, and vi) *Proximity* edges with *AtLocation* and *UsedFor* edges. We report the main Localisation Success Rate (LSR) measure for all variants, as well as the scene average percentage of object nodes which are linked by 0, 1, or 2 types of semantic edges, i.e., *AtLocation* and *UsedFor* edges.

Discussion. Table II shows that *AtLocation* is more effective than *UsedFor* for localising objects. This is reasonable, since the *AtLocation* edge leads to message passing among objects that are connected in the same location, containing information more relevant to the localisation task. However, the best performance is obtained when the **D-SCG** rely on all types of edges which provides a higher connectivity among object nodes to concept nodes. There are 80% object nodes linked to concept nodes by

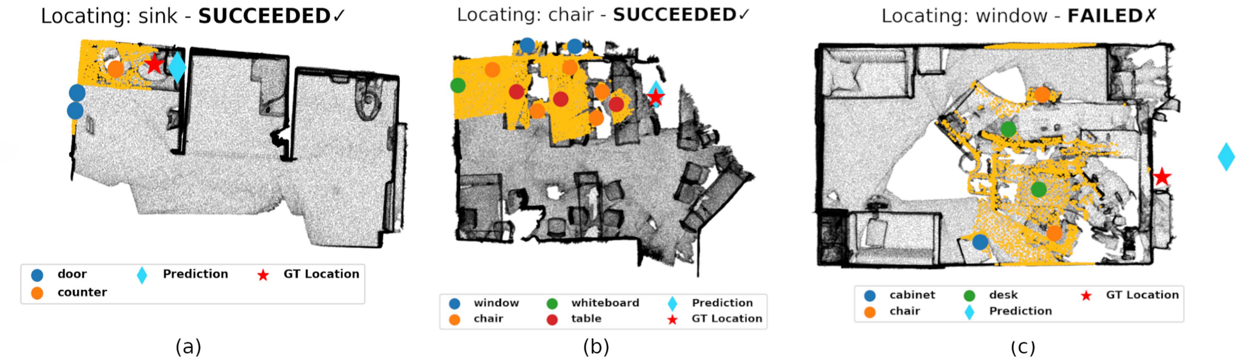


Fig. 8. Qualitative results obtained with **D-SCG-OL**. The partially known scene is coloured with a yellow background, while the unknown scene is indicated with grey. The coloured circles indicate the object nodes present in the **D-SCG**. The red star indicates the GT position of the target object, while the cyan diamond indicates the predicted positions. The network is able to correctly predict the position of a sink in (a) and a chair in (b). In the failure case of (c), the network correctly identified the direction of the window but overestimated the distance from the visible objects.

both *AtLocation* and *UsedFor* edges, leading to a more effective knowledge fusion than when only one type of semantic edge is used.

Which attention network is more effective? We examine the usefulness of the proposed attention mechanism in **D-SCG-OL** compared to other, commonly used, attentional message passing modules in the GNN literature. As most of these approaches do not support the use of edge features, we modify the node features for this ablation study to include the positional information to the node features. For a fair comparison, we remove the edge embedding from **D-SCG-OL**. The set of attention networks we compare with is listed below:

- *No attention [38]* is the first baseline, where we use GraphSAGE without relying on any attention module.
- *GAT [39]* adds an attention mechanism to the message passing procedure.
- *GATv2 [40]* is similar to GAT but improves the attention mechanism in terms of the expressiveness and addresses the problem of "static attention" when using GATs for message passing.
- *HAN [41]* defines multiple meta-paths that connect neighbouring nodes either by specific node or edge types. It employs attentional message passing sequentially by first calculating the semantic-specific node embedding and then updating them by using an attention mechanism [36]. With **D-SCG** we define three sets of meta neighbours, i.e., the proximity neighbours, the *AtLocation* neighbours, and the *UsedFor* neighbours, connected by the specific edges. We implement the message passing for each meta-path using specialised GraphTransformer layers.
- *GraphTransformer [29]* is similar to ours, except that it does not accommodate sparse attention and has less expressive power due to the smaller number of parameters. This module is essentially a porting of the scaled dot-product attention mechanism [36] to GNNs.
- *Ours - Only ReLA [5]* is our GNN with the original ReLA [5], i.e., without the ScaleNorm and Linear layer.

Discussion. As shown in Table III, different attention modules can produce results that vary greatly in terms of LSR. Among all, HAN achieves the worst performance, showing that features are better to be propagated simultaneously rather than sequentially.

TABLE III
IMPACTS OF DIFFERENT ATTENTION MODULES FOR THE OBJECT LOCALISATION TASK WITH OUR **D-SCG-OL**. LSR: LOCALISATION SUCCESS RATE

Graph Attention Type	LSR ↑
No attention [38]	0.199
GAT [39]	0.179
GATv2 [40]	0.202
HAN [41]	0.137
GraphTransformer [29]	0.187
Ours - Only ReLA [5]	0.190
Ours	**0.215**

TABLE IV
IMPACT OF DIFFERENT NUMBERS OF MESSAGE PASSING LAYERS IN OUR **D-SCG-OL**. LSR: LOCALISATION SUCCESS RATE

# Layers	1	2	3	4	5
LSR ↑	0.180	0.257	0.283	**0.297**	0.285

TABLE V
COMPARISON OF OBJECT LOCALISATION PERFORMANCE IN THE 3D ENVIRONMENT INSTEAD OF ON THE 2D FLOOR PLANE

Method	LSR ↑
LayoutTransformer [34]	0.158
SCG-OL [4]	0.048
D-SCG-OL	**0.258**

GAT and GraphTransformer perform better than HAN, yet it is still worse than GraphSAGE which uses no attention. This is potentially due to the limitations of the standard attention mechanism when used in GNNs [40]. GraphSAGE is a general inductive framework that leverages node feature information at different depths and is proven to work well on large graphs. In general the attention module should be carefully designed in order to provide advantageous performance. For example, GATv2 improves the localisation performance by fixing the static attention problem of the standard GAT.

Our ReLA-based attention model avoids the usage of a softmax as in the original Graph Transfomer [29] achieves the

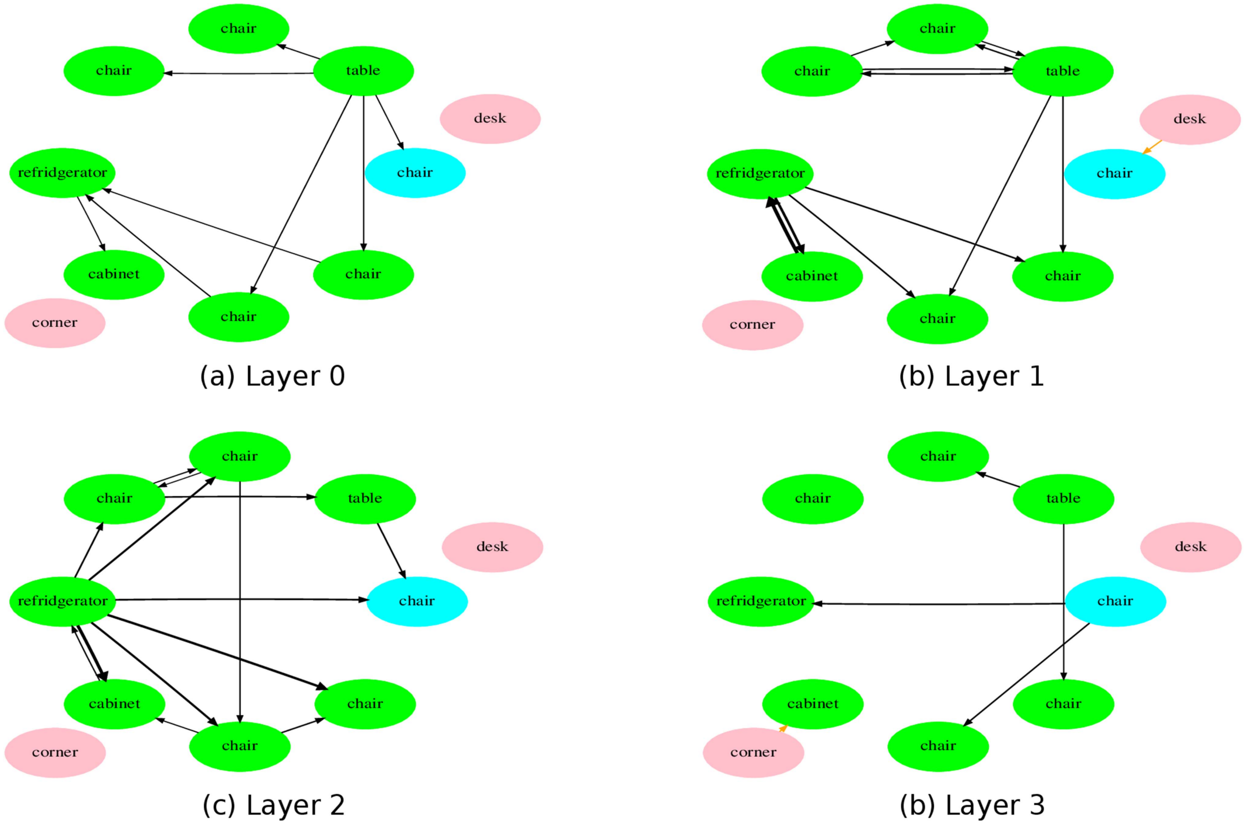


Fig. 9. Feature propagation at different layers of our GNN that are directed by our attention module. The cyan node indicates the target object, the green nodes represent the scene nodes, and the pink nodes represent the concept nodes. The black edges indicate the sharing of information between two nodes in the direction indicated by the arrows. For ease of visualisation, we show edges with a mean attention weight over the heads that are superior to 0.2%, and only display concept nodes that are connected via these types of edges.

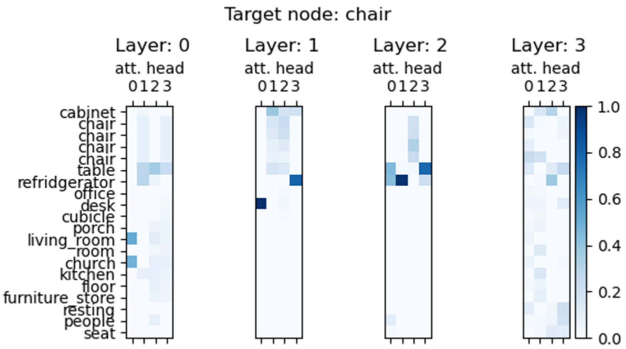


Fig. 10. Attention weights for messages that are propagated to the target node are indicated in Fig. 9. The network learns to propagate information from different nodes by leveraging different attention heads. The first and last layer the network propagate information from most of the neighbouring nodes, while the intermediate layers focus on few specific nodes.

best overall performance in terms of LSR. This substantial improvement is contributed by the increased expressive power and the ability to reason on sub-graphs during the message passing procedure. The usage of only ReLA, i.e., without scale normalisation and the successive re-projection [31] achieves a lower LSR compared to ours, confirming the advantage brought by the proposed additional normalisation as the learned weights are positively unbounded.

Does the number of message passing layers and the final node concatenation of ***D-SCG-OL*** *make a difference?*

We examine a set of variants of our **D-SCG-OL** with varying numbers of message passing layers ranging from 1 to 5. Table IV shows that using four message passing (MP) layers leads to the best performance. When using a single MP layer, there is not enough information regarding the context to be propagated to the nodes and this leads to the worst performance. With more than two MP layers, the performance starts to increase, saturating at four layers. With additional layers, we observe that the performance starts to degrade. This might be due to the over-smoothing problem [42], [43], where after multiple message passing rounds, the embeddings for different nodes are indistinguishable from one another. Given the best layer number, we also validate the choice of concatenating the original embedding to the aggregated *contextual* ones, instead of using only the aggregated features. Concatenation is more advantageous with an LSR of 0.29, while directly using the aggregated node representation leads to an LSR of 0.28. We argue that this happens because concatenation allows the network to still remember the initial representation, developing a better understanding of the context after message passing.

Localising in 3D. We examine the network capability to localise the target object directly in 3D scenes instead of on the 2D floor plane. We compare **D-SCG-OL** with **SCG-OL** by making the appropriate modifications for 3D localisation.

Table V reports the localisation performance in the 3D scenes. We can observe that all the three methods suffer a drop in terms of LSR performance due to the increased difficulty level of the problem. **SCG-OL** experiences the highest drop in performance, with an LSR score of only 0.05, down from its original score of 0.24 when evaluated in the 2D domain. This decrease in performance can be attributed to a difficulty in representing the object arrangement using only distances when an additional dimension is considered. Utilising a less abstract representation by using relative positions between objects leads to much more accurate results. Despite the increased problem difficulty, our proposed **D-SCG-OL** achieves the best LSR of 0.26, which is significantly higher than the second-best method LayoutTransformer with a LSR of 0.15.

Attention Visualisation In Figs. 9 and 10 we show how the network prioritises the exchange of information when localising a chair. Note that our network does not use the softmax function when calculating the attention weights, thus they do not necessarily sum to one. We normalise the weights for the visualisation results. Fig. 9 shows the features propagated via message passing that are assigned a high weight by our attention modules. The network learns to operate very differently depending on the layer, and most of the attention weights are given to edges between object nodes. The network also learns to attend differently to instances of the same object based on the scene geometry that is described by the edge features. For instance, in the first layer only two of the five chairs nodes propagate their features with a high weight to the refrigerator. Incidentally, these nodes represent the two chairs closest to the fridge. Fig. 10 shows the different heads' attention scores for messages that are propagated to the target node. We can see that each head focuses on different nodes: some heads are giving high weights to specific nodes, e.g., head zero and three of the second layer, while others balance the features from many nodes, e.g., head two and three of the first layer. Lastly, we can see that most of the commonsense information is propagated in the first and last layer of the GNN.

VI. Conclusion

We proposed an novel scene graph model, the **D-SCG**, to address the problem of localising objects in a partial 3D scene. The spatial information regarding the arrangement of the object is described via directional edges with relative positions instead of undirectional relative distances as in the prior work. With the proposed **D-SCG**, we developed a new GNN-based solution for object localisation, **D-SCG Object Localiser**, that can directly estimate the position of the target object by predicting its relative positions with respect to other objects in the partially observed scene, leading to an efficient end-to-end trainable solution. Our approach also features a new attention module, wrt to our previous approach, to further improve the localisation performance, by using the ReLA attention. We thoroughly evaluated our proposed method on the partial scene dataset and proved its superior performance in terms of localisation success rate against baselines and the state-of-the-art methods. Finally, we showed that our approach can be applied for 3D object localisation with a marginal performance drop, while the previous state-of-the-art method degrades dramatically due the increased localisation difficulty. Future work will focus on scaling our proposed approach to large-scale outdoor scenarios and extending to robotic applications.

References

[1] D. Batra et al., "Objectnav revisited: On evaluation of embodied agents navigating to objects," 2020, *arXiv:2006.13171*.

[2] A. Luo, Z. Zhang, J. Wu, and J. B. Tenenbaum, "End-to-end optimization of scene layout," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 3753–3762.

[3] R. Speer, J. Chin, and C. Havasi, "ConceptNet 5.5: An open multilingual graph of general knowledge," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 4444–4451.

[4] F. Giuliari, G. Skenderi, M. Cristani, Y. Wang, and A. Del Bue, "Spatial commonsense graph for object localisation in partial scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 19496–19505.

[5] B. Zhang, I. Titov, and R. Sennrich, "Sparse attention with linear units," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2021, pp. 6507–6520.

[6] Y. Liu, R. Wang, S. Shan, and X. Chen, "Structure inference net: Object detection using scene-level context and instance-level relationships," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6985–6994.

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

[8] S. Schuster, R. Krishna, A. X. Chang, L. Fei-Fei, and C. D. Manning, "Generating semantically precise scene graphs from textual descriptions for improved image retrieval," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 70–80.

[9] N. Xu, A.-A. Liu, J. Liu, W. Nie, and Y. Su, "Scene graph captioner: Image captioning based on structural visual representation," *J. Vis. Commun. Image Representation*, vol. 58, pp. 477–485, 2019.

[10] X. Yang, K. Tang, H. Zhang, and J. Cai, "Auto-encoding scene graphs for image captioning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10677–10686.

[11] J. Gu, S. Joty, J. Cai, H. Zhao, X. Yang, and G. Wang, "Unpaired image captioning via scene graph alignments," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 10322–10331.

[12] J. Shi, H. Zhang, and J. Li, "Explainable and explicit visual reasoning over scene graphs," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8368–8376.

[13] S. Lee, J.-W. Kim, Y. Oh, and J. H. Jeon, "Visual question answering over scene graph," in *Proc. 1st Int. Conf. Graph Comput.*, 2019, pp. 45–50.

[14] P. Gay, J. Stuart, and A. Del Bue, "Visual graphs from motion (VGfM): Scene understanding with object geometry reasoning," in *Proc. Asian Conf. Comput. Vis.*, 2018, pp. 330–346.

[15] J. Wald, H. Dhamo, N. Navab, and F. Tombari, "Learning 3D semantic scene graphs from 3D indoor reconstructions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 3960–3969.

[16] S.-C. Wu, J. Wald, K. Tateno, N. Navab, and F. Tombari, "SceneGraphFusion: Incremental 3D scene graph prediction from RGB-D sequences," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 7515–7525.

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

[18] Y. Zhou, Z. While, and E. Kalogerakis, "SceneGraphNet: Neural message passing for 3D indoor scene augmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 7383–7391.

[19] K. Wang, Y.-A. Lin, B. Weissmann, M. Savva, A. X. Chang, and D. Ritchie, "PlanIT: Planning and instantiating indoor scenes with relation graph and spatial prior networks," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 132–147, 2019.

[20] Z. Ravichandran, L. Peng, N. Hughes, J. D. Griffith, and L. Carlone, "Hierarchical representations and explicit memory: Learning effective navigation policies on 3D scene graph using graph neural networks," in *Proc. Int. Conf. Robot. Automat.*, 2022, pp. 9272–9279.

[21] D. Vrandečić and M. Krötzsch, "Wikidata: A free collaborative knowledgebase," *Commun. ACM*, vol. 57, pp. 78–85, 2014.

[22] K. Faldu, A. Sheth, P. Kikani, and H. Akabari, "KI-BERT: Infusing knowledge context for better language and domain understanding," 2021, *arXiv:2104.08145*.

[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics*, 2019, pp. 4171–4186.

[24] J. Bao, N. Duan, M. Zhou, and T. Zhao, "Knowledge-based question answering as machine translation," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2014, pp. 967–976.

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

[26] A. Narayanan, A. Rao, A. Prasad, and N. S., "VQA as a factoid question answering problem: A novel approach for knowledge-aware and explainable visual question answering," *Image Vis. Comput.*, vol. 116, 2021, Art. no. 104328.

[27] J. Gu, H. Zhao, Z. L. Lin, S. Li, J. Cai, and M. Ling, "Scene graph generation with external knowledge and image reconstruction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1969–1978.

[28] R. Speer and J. Lowry-Duda, "ConceptNet at SemEval-2017 task 2: Extending word embeddings with multilingual relational knowledge," in *Proc. Int. Workshop Semantic Eval.*, 2017, pp. 85–89.

[29] Y. Shi, Z. Huang, S. Feng, H. Zhong, W. Wang, and Y. Sun, "Masked label prediction: Unified message passing model for semi-supervised classification," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. 1548–1554.

[30] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.

[31] T. Q. Nguyen and J. Salazar, "Transformers without tears: Improving the normalization of self-attention," in *Proc. 16th Int. Conf. Spoken Lang. Transl.*, 2019.

[32] 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/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2432–2443.

[33] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," 2018, *arXiv:1801.09847*.

[34] K. Gupta, J. Lazarow, A. Achille, L. S. Davis, V. Mahadevan, and A. Shrivastava, "Layouttransformer: Layout generation and completion with self-attention," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 984–994.

[35] C. Ying et al., "Do transformers really perform badly for graph representation?," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021, pp. 28877–28888. [Online]. Available: https://openreview.net/forum?id=OeWooOxFwDa

[36] A. Vaswani et al., "Attention is all you need," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[37] N. Shazeer and M. Stern, "Adafactor: Adaptive learning rates with sublinear memory cost," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4603–4611.

[38] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.

[39] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Representations*, 2018.

[40] S. Brody, U. Alon, and E. Yahav, "How attentive are graph attention networks?," in *Proc. Int. Conf. Learn. Representations*, 2022.

[41] X. Wang et al., "Heterogeneous graph attention network," in *Proc. World Wide Web Conf.*, 2019, pp. 2022–2032.

[42] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 3438–3445.

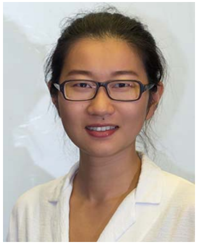
[43] K. Oono and T. Suzuki, "Graph neural networks exponentially lose expressive power for node classification," in *Proc. Int. Conf. Learn. Representations*, 2020.

Francesco Giuliari (Student Member, IEEE) received the MsC degree in computer science from the University of Verona, in 2018. He is currently working toward the PhD degree with the University of Genoa. He is currently affiliated with Istituto Italiano di Tecnologia under the supervision of Dr. Alessio Del Bue. His main research interests include computer vision, scene understanding, and vision-based agent navigation.

Geri Skenderi received the master's degree in computational data science from the Free University of Bolzano-Bozen, in 2020. He is currently working toward the PhD degree with the University of Verona, working under the supervision of Prof. Marco Cristani. His research interests cover the broad area of graph learning and disentangled representation learning, with an applicative focus on forecasting and prediction.

Marco Cristani (Member, IEEE) is full professor (Professore Ordinario) with the Computer Science Department, University of Verona, associate member with the National Research Council (CNR), external collaborator with the Italian Institute of Technology (IIT). His main research interests include statistical pattern recognition and computer vision, mainly in deep learning and generative modeling, with application to social signal processing and fashion modeling. On these topics, he has published more than 180 papers, including two edited volumes, 46 international journal papers, 126 conference papers and 13 book chapters. He has organized 11 international workshops, cofounded a spin-off company, Humatics, dealing with e-commerce for fashion. He is or has been principal investigator of several national and international projects, including PRIN and H2020 projects. He is a member of the editorial board of the *Pattern Recognition* and *Pattern Recognition Letters* journals. He is managing director of the Computer Science Park, a Technology Transfer Center, University of Verona. He is a member of the ACM and IAPR.

Alessio Del Bue (Member, IEEE) is a tenured senior researcher leading the Pattern Analysis and computer VISion (PAVIS) Research Line of the Italian Institute of Technology (IIT), Genoa, Italy. He is a coauthor of more than 100 scientific publications in refereed journals and international conferences on computer vision and machine learning topics. His current research interests include 3D scene understanding from multi-modal input (images, depth, and audio) to support the development of assistive artificial intelligence systems. He is a member of the technical committees of major computer vision conferences (CVPR, ICCV, ECCV, and BMVC). He serves as an associate editor of the *Pattern Recognition* and *Computer Vision and Image Understanding* journals. He is a member of the ELLIS.



Yiming Wang received the PhD degree in electric engineering from the Queen Mary University of London, U.K., in 2018, working on vision-based multi-agent navigation. She is a researcher with the Deep Visual Learning (DVL) Unit, Fondazione Bruno Kessler (FBK). Her research mainly focuses on vision-based scene understanding that facilitates automation for social good. Since 2018, she has worked as a post-doc researcher with the Pattern Analysis and Computer Vision (PAVIS) Research Line, Istituto Italiano di Tecnologia (IIT), working on topics related to active 3D vision. She is actively serving as a reviewer for top-tier conferences and journals in both the computer vision and robotics domains.

Open Access provided by 'Istituto Italiano di Tecnologia' within the CRUI CARE Agreement