Embodied Semantic Scene Graph Generation

Anonymous Author(s)
Affiliation
Address
email

Abstract: Semantic scene graph provides an effective way for intelligent agents to better understand the environment and it has been extensively used in many robotic applications. Existing work mainly focuses on generating the scene graph from the sensory information collected from a pre-defined path, while the environment should be exhaustively explored with a carefully designed path in order to obtain a comprehensive semantic scene graph efficiently. In this paper, we propose a new task of Embodied Semantic Scene Graph Generation, which exploits the embodiment of the intelligent agent to autonomously generate an appropriate path to explore the environment for scene graph generation. To this end, a learning framework with the paradigms of imitation learning and reinforcement learning is proposed to help the agent generate proper actions to explore the environment and the scene graph is incrementally constructed. The proposed method is evaluated on the AI2Thor environment using both the quantitative and qualitative performance indexes. Additionally, we implement the proposed method on a streaming video captioning task and promising experimental results are achieved.

Keywords: Semantic Scene Graph, Embodied Exploration, Learning for Visual Navigation

1 Introduction

The scene graph is a collection of nodes in a graph structure where nodes usually represent scene entities and edges represent geometrical transformation between the nodes. It provides a flexible way to track objects and their spatial relations within the scene. Recently, the concept of scene graph is successfully adapted in many applications in computer vision and robotics [1][2]. For example, the Visual Genome dataset [3], which contains annotations of objects and their relationships in images, has been extensively utilized in various scene graph generation tasks [4][5][6]. In [1], the author proposes to extract the knowledge graph from recorded videos for better video understanding. Ref.[2] provides an extensive survey on the recent progress on the generation and applications of scene graph.

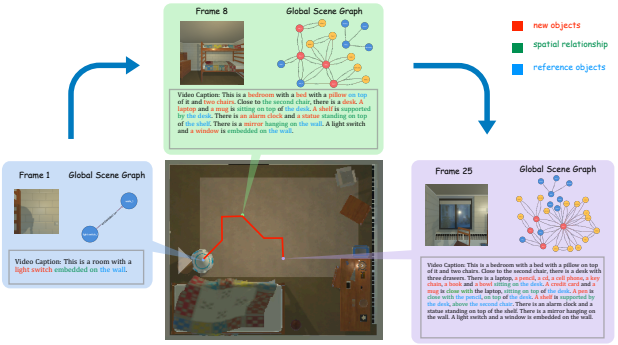

Figure 1: The illustration for the Embodied Semantic Scene Graph Generation. The agent starts to move and always selects an appropriate action to obtain better semantic scene graph. Further, the obtained semantic scene graph can be helpful to generate the streaming video captioning.

Furthermore, 3D semantic scene graph is of great interest for robotic applications such as navigation, mapping, and interaction [7]. To understand the 3D environment, Ref.[8] proposes a 3D scene graph construction framework using RGB-D data processing pipeline, which can be used for task

planning and visual question answering. Ref.[9] develops a Boltzmann Machines-based generative scene model bringing objects, their spatial relations and affordances together. Ref.[10] presents a 3D scene graph, which unifies multi-modal semantic information in the 3D space. Recently, [11] proposes 3D dynamic scene graph which extends to represent dynamic scenes with moving agents.

Though great success has been achieved for semantic scene graph, all of the above work focuses on generating scene graph from the sensory information collected from a pre-defined path. However, to obtain a comprehensive and informative semantic scene graph, the environment should be exhaustively explored, which requires large time consumption. In many scenarios, the disconnection between the scene graph generation and the exploration path even results in unsatisfied results. Therefore, it is necessary for the agent to have the ability to autonomously explore the environment to generate a good semantic scene graph.

In this work, we adopt the embodied exploration framework to tackle the *Embodied Semantic Scene Graph Generation* problem, which has never been addressed before. It combines the action and perception abilities of the agent. The embodied agent generates actions to autonomously explore the environment and then incrementally constructs a high-quality semantic scene graph using the detected object instances and contextual information. The collected sensor data is continuously processed and incorporated into the scene graph model along with the exploration process. The illustration is demonstrated in Fig.1. Different from the existing work which focuses on generating scene graphs from acquired sensor information, this work introduces the embodiment of the agent to effectively combine the action, vision and language together for autonomous 3D semantic scene graph generation. The main contributions are summarized as follows:

1. We propose a new framework for the Embodied Semantic Scene Graph Generation problem, which exploits the embodiment characteristic of the intelligent agent to find an appropriate path for scene graph generation in an embodied environment.
2. We develop a learning framework with paradigms of imitation learning and reinforcement learning to help the agent acquire the intelligence to generate high-quality scene graph.
3. We testify the proposed method on the AI2Thor dataset and evaluate its effectiveness using the quantitative and qualitative performance indexes.
4. We implement the proposed method on a streaming video captioning task, which shows promising experimental results.

For the rest of the paper, Section II presents the related work. Problem formulation and architecture is instroduced in Section III. Sections IV and V give details about the scene graph generation and navigation modules. Section VI presents the experimental results and Section VII closes this paper.

2 Related Work

This work falls into the intersection of embodied exploration and semantic understanding. Therefore, we present a brief review on both domains.

The embodied exploration lies in the intersection of computer vision and robotics. The representative work includes embodied question answering [12], embodied captioning [13], embodied amodal detection [14], multi-agent embodied exploration [15], etc. In those work, one or multiple embodied agents equipped with visual perception modules can autonomously explore the environment to perform various active perception tasks. Furthermore, [16] studies the task of embodied interactive learning for object detection. It should be noticed that in [13], the authors propose an embodied captioning task, which is relevant to our work, while our work solves a more fundamental task of generating the 3D scene graph, which provides more comprehensive contextual information of the environment. Actually, the generated scene graph can be useful for many down-streaming tasks such as image captioning, streaming video captioning, embodied Questions and Answering, etc.

On the other hand, the embodied exploration task can also benefit from the semantic scene graph. Refs.[17] and [18] study how to enhance the active object search using the spatial contexts and semantic linking maps respectively. The hierarchical mechanical search also shows performance improvement by using the semantic modeling [19]. In [20], the authors propose to learning object relation graph and tentative policy for visual navigation. Besides, the scene graph also plays an important role in the visual question answering system for robot manipulation [21]. These works

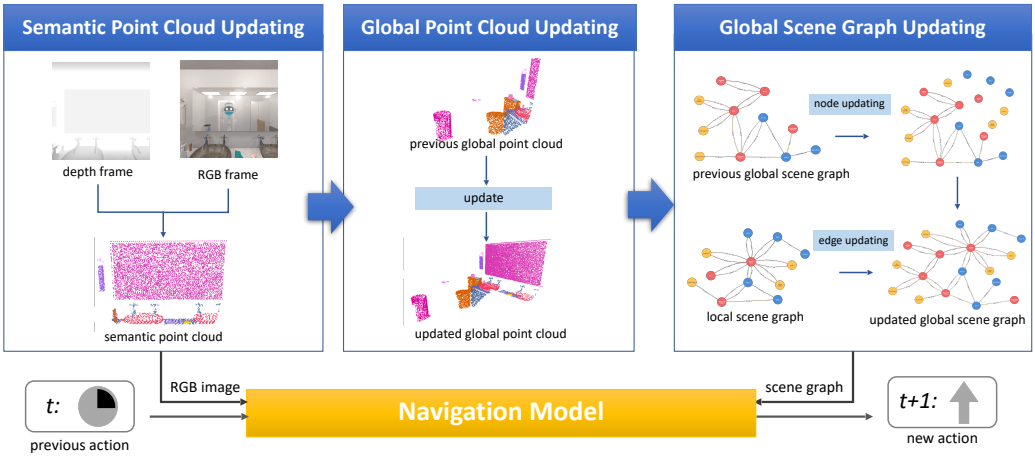

Figure 2: The architecture of the *Embodied Semantic Scene Graph Generation*. At time instant t, after taking one action, the agent comes to a new viewpoint and updates the local and global scene graph, and further exploits the local and global scene graph to generate a new action. This process iterates until a *stop* action is obtained.

illustrate the significance of semantic scene graph in embodied perception. However, they all use existing scene graphs and do not address the generation of the scene graph.

3 Problem Formulation and Architecture

The goal of this work is to enable the agent to automatically generate a sequence of actions to explore the environment and build the corresponding semantic scene graph incrementally. Concretely speaking, the agent starts from any location of a scene and initial empty global semantic scene graph $\mathcal{GSSG}_0$. At time instant t, the agent could construct a local scene graph $\mathcal{LSSG}_t = \{\mathcal{N}_t^{(l)}, \mathcal{R}_t^{(l)}\}$ from its observation s_t, which could be extracted from the observed RGB and depth images. The local scene graph $\mathcal{LSSG}_t$ should be merged with the previous global scene graph $\mathcal{GSSG}_{t-1} = \{\mathcal{N}_{t-1}^{(g)}, \mathcal{R}_{t-1}^{(g)}\}$ to get the updated global scene graph $\mathcal{GSSG}_t = \{\mathcal{N}_t^{(g)}, \mathcal{R}_t^{(g)}\}$. Please note that we follow the conventional symbols with slight modification [22], i.e. $\mathcal{N}_t^{(l)}$ and $\mathcal{R}_t^{(l)}$ represent the node and semantic relationships for the local scene graph $\mathcal{LSSG}_t$, and $\mathcal{N}_t^{(g)}$ and $\mathcal{R}_t^{(g)}$ represent the node and semantic relationships for the global scene graph $\mathcal{GSSG}_t$. After that, a navigation module should produce an action a_t for the agent to obtain the next observation s_{t+1}. The above procedure iterates until a satisfactory global semantic scene graph is achieved and the stop action is triggered. The obtained semantic scene graph can then be used for describing the environment comprehensively and some down-streaming tasks such as captioning, question answering, etc.

We call this problem as *Embodied Semantic Scene Graph Generation*, which exploits the embodiment capability of the agent to collect data and construct high-quality scene graph. It is significantly different from existing scene graph generation work such as [8][10]. To solve this problem, we design an architecture as shown in Fig.2. It is mainly composed of the scene graph generation and navigation module. For the scene graph generation, a 3D semantic point cloud is produced from the RGB and depth images and is used for generating the local scene graph, which is then used to incrementally update the global scene graph. For the navigation module, we utilize the RGB frame, previous action and the scene graph as input and an action can be generated for the next step.

4 Scene Graph Generation

The scene graph generation module can be divided into two parts: local scene graph prediction and global scene graph generation, which are introduced as follows.

Local scene graph prediction: we mainly follow the scene graph generation method proposed in [23][22] with slight modification to predict the local scene graph. The generated local semantic scene graph includes the current in-sight objects and their relations with each other. We additionally introduce object class embedding and bounding box coordinates as the input for each object node, and the training objective is to restore the bounding box coordinates and label for each object and edge. Please see the supplementary material for more details.

Global scene graph generation: At each time step, the agent takes an action to move, and the equipped camera captures the RGB and depth frames of the scene to construct a local scene graph, which is further merged into the global scene graph. During this period, the detected objects in the local scene graph and the existing objects in the previous global scene graph is aligned.

Different from the *multi-view consistency* considered in [10], which has the 3D mesh and multi-view visual information known in advance, we do not have any prior information about the scene and the information of each object is incrementally updated. Considering this situation, we propose a point cloud based weighted voting mechanism which utilizes the number of the pixels in the point cloud to measure the confidence of the object category. In practice, for each detected object, we maintain a score distribution across predefined classes. The score is incrementally accumulated by the weighted value of the confidence provided by the detector and the size of the acquired point cloud.

Then, we align the object point cloud with the nodes in global scene graph by computing the fraction of point cloud in the object point cloud that is inside the node's 3D bounding box. We calculate this fraction between each object point cloud and the node in the global scene graph. If the highest fraction of object point cloud and the corresponding node is higher than the preset threshold and this fraction is higher than those with other objects, the object and the node is considered aligned. This bidirectional alignment procedure could ensure that each aligned object and the corresponding node are in one-to-one correspondence. Fig.3 demonstrates the alignment process. Based on the aligning result for each object and the weighted voting mechanism, we could align the nodes and edges in the obtained Local Semantic Scene Graph(LSSG) with those in the Global Semantic Scene Graph(GSSG). The updating operation could involve ADD, UPDATE and REPLACE for both nodes and edges in GSSG. Please see the supplementary material for more details.

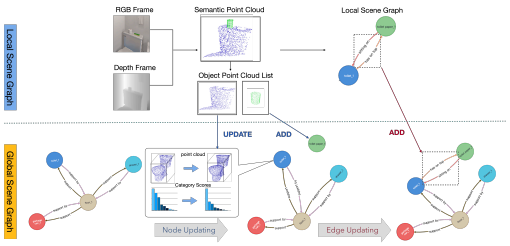

Figure 3: The alignments between objects and edges of local scene graph and global scene graph.

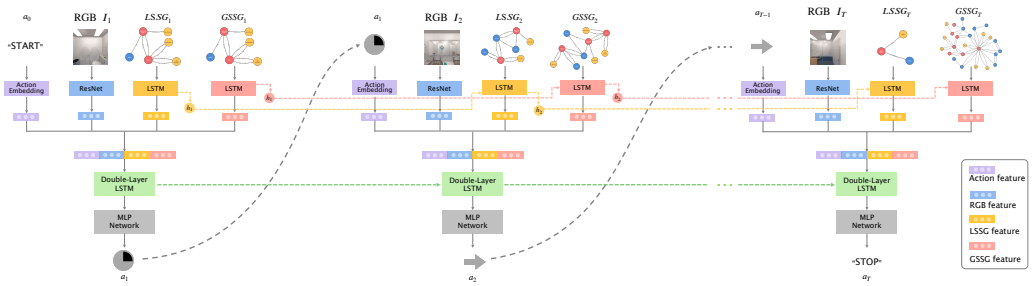

Figure 4: The architecture of the navigation model.

5 Navigation Model

5.1 Model Structure

The goal of the proposed navigation model is to guide the agent to take actions to explore the environment and build the sematic scene graph incrementally. The action space considered in this work includes *Move*, where the agent can take nine basic actions corresponding to 8 directions and

no move, and *Rotate*, where the agent can rotate clockwise or counterclockwise with a fixed angle $\delta_r = 90°$ or no rotate. Especially, if no move and no rotate is selected, a stop action is triggered.

The design of the navigation model is a challenging problem since the agent is expected to build a good semantic scene graph which contains the objects and their relationships accurately. To tackle this problem, we construct a navigation architecture which is shown in Fig.4. We use a ResNet to extract the visual feature, two LSTMs to encode the local scene graph and global scene graph sequence respectively, and an embedding module to represent the previous action. The three feature vectors are concatenated and fed into a double-layer LSTM model. Finally, an MLP network is used to obtain the probability distribution for the next step action.

5.2 Learning for Navigation

Since the direct learning for such an architecture is still challenging, we adopt the imitation learning as the pre-training and the reinforcement learning for fine-tuning.

5.2.1 Imitation Learning

The goal of imitation learning for sequential prediction problem is to train the agent to mimic the expert's behaviors. In this work, it is important to generate some demonstration paths for the agent to imitate. We adopt a two-stage method to deal with this problem. In the first stage, we try to obtain the waypoint set and in the second stage, we perform the interpolation between way-points to get the whole demonstration path. To evaluate the way-points, we first count the visible object at each viewpoint, and select the closest next way-point that has the most number of unseen new objects, which can be expressed as:

$$v^* = \underset{v \in O(v_c, k^*)}{\arg\max}\ new_object_num(v),\ k^* = \arg\min_{k} \sum_{v \in O(v_c, k)} new_object_num(v) > 0$$

and the way-point set can be updated as $\mathcal{W} = \mathcal{W} \cup \{v^*\}$. The above procedure is repeated until a maximum distance is achieved. In the second step, we implement a beam search over the key point sequence for interpolation [24]. With the obtained demonstration paths, we use the negative log likelihood loss to train our agent.

5.2.2 Reinforcement Learning

After pre-training the navigation model with imitation learning, we then try to further improve its performance using the REINFORCE algorithm. Since our task aims at formulating a scene graph for the entire scene, we directly measure the similarity between the generated scene graph and the ground truth one. Given a $\mathcal{GSSG}$ which is constructed from one path, the similarity score can then be calculated as a weighted sum of the precision and recall rate of the nodes and edges. On the other hand, we hope to encourage the observation diversity when constructing the scene graph. The diversity can be characterized as the number of observation viewpoints of the detected objects. Finally, the reward is designed considering the similarity score and object diversity. Please see the details of reward design and implementation in supplementary material.

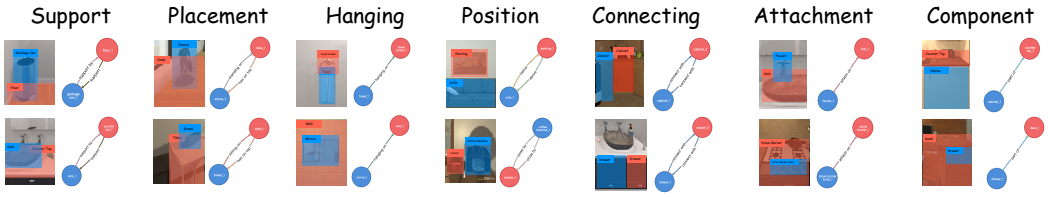

Figure 5: An illustration of the considered semantic relationships.

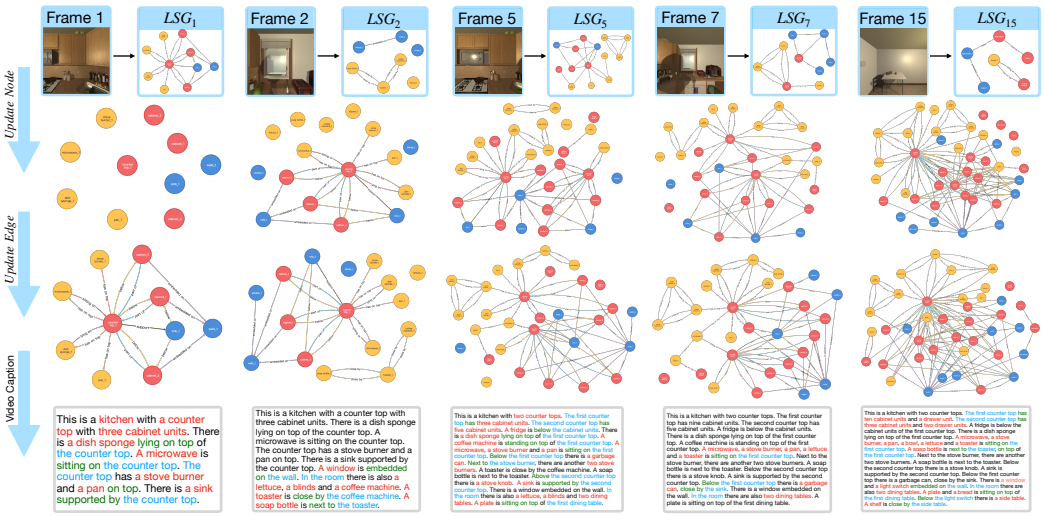

Figure 6: A representative result for our *embodied semantic scene graph generation* framework.

6 Experimental Validation

6.1 Dataset

We generate a new dataset with the AI2THOR [25], which contains 120 scenes utilizing the object attribute information. We follow the guidelines in [22] to extract 16 semantic relationships, which can be clustered into the following seven categories (Fig.5). Please refer to the supplementary material for more details.

6.2 Method and Metrics

To verify the effectiveness of our framework and the influence of the input vectors, we introduce the following methods for comparison:

1. **Random** The agent always randomly selects an action from the action space to perform the scene exploration.

2. **PPO+Map[26]** The agent utilizes Resnet-18 to encode RGB frame, local map and global map respectively and concatenate them as the input for the LSTM to predict the next action.

3. **Frontier-based Exploration[27]** A heuristic algorithm that guides the agent to detect and sweep to the closest frontier with path planning.

4. **LSTM+Act+Pose[12, 13]** This model is widely used in embodied tasks, it encodes RGB frame, agent position and last step action as the input of LSTM to predict the next step action for the agent.

5. **LSSG+GSSG(Ours)** Our framework that further takes the LSSG and GSSG vectors as the input of LSTM.

6. **Traverse(Oracle)** Ask the agent to traverse every viewpoint in the scene.

To evaluate the quality of the semantic scene graph, we adopt the precision, recall rate and F1 score of the nodes and edges compared to the ground truth scene graph, denoted as P_{node}, R_{node}, $F1_{node}$, P_{edge}, R_{edge} and $F1_{edge}$ respectively, and the number of steps NoS as quantitative metrics. Note that R_{node} could be viewed as an object level coverage, and R_{edge} could represent the semantic richness of the generated $\mathcal{GSSG}$.

We also propose the following two tasks for evaluation. **RelCls:** In this task, we align the detecting result from Mask RCNN with ground truth to obtain the correctly detected objects at each viewpoint, and offer the ground truth segmentation of each object to the agent. Since the detection results and point cloud information are always correct, the precision of node is always 1.0. **SGGen:** In this

Table 1: Performance Comparison

	RelCls							SGGen						
	Nos	P_{node}	R_{node}	$F1_{node}$	P_{edge}	R_{edge}	$F1_{edge}$	NoS	P_{node}	R_{node}	$F1_{node}$	P_{edge}	R_{edge}	$F1_{edge}$
Random	27.367	1.0	0.311	0.474	0.705	0.144	0.239	27.367	0.550	0.223	0.317	**0.278**	0.097	0.144
PPO+Map	23.915	1.0	0.484	0.652	**0.736**	0.260	0.384	23.915	**0.665**	0.435	0.526	0.248	0.200	0.221
Frontier-based	24.597	1.0	0.547	0.707	0.717	0.289	0.412	24.597	0.643	0.425	0.512	0.253	0.194	0.220
LSTM+Act+Pose	22.815	1.0	0.563	0.720	0.689	0.300	0.418	22.815	0.567	0.526	0.546	0.235	0.208	0.221
LSSG+GSSG(Ours)	22.435	1.0	**0.688**	**0.815**	0.681	**0.339**	**0.453**	26.657	0.558	**0.564**	**0.561**	0.228	**0.230**	**0.229**
Traverse(Oracle)	>500	1.0	0.858	0.924	0.576	0.384	0.461	>500	0.516	0.617	0.562	0.201	0.275	0.232

Table 2: Ablation Study

	RelCls							SGGen						
	Nos	P_{node}	R_{node}	$F1_{node}$	P_{edge}	R_{edge}	$F1_{edge}$	NoS	P_{node}	R_{node}	$F1_{node}$	P_{edge}	R_{edge}	$F1_{edge}$
Base	22.815	1.0	0.563	0.720	0.689	0.300	0.418	22.815	0.567	0.526	0.546	**0.235**	0.208	0.221
GSSG	17.254	1.0	0.577	0.732	**0.694**	0.309	0.428	22.942	0.524	0.460	0.490	0.228	0.200	0.213
LSSG	23.778	1.0	0.657	0.793	0.678	0.337	0.450	22.829	0.541	0.539	0.540	0.229	0.213	0.224
LSSG+GSSG	22.435	1.0	**0.688**	**0.815**	0.681	**0.339**	**0.453**	26.657	0.558	**0.564**	**0.561**	0.228	**0.230**	**0.229**

task, the agent utilizes the raw detecting result from Mask RCNN which is consistent with real-world environment. The agent is expected to leverage both the global scene graph and navigation policy to deal with the imperfection of Mask RCNN and the incompleteness of the collected point cloud, incrementally generating an accurate and informative scene graph.

6.3 Results

6.3.1 Quantitative results

The experimental results are shown in Table I and II, from which we have the following observations:

1. In both RelCls and SGGen, $LSSG + GSSG$ outperforms other baselines on the recall rate and F1-score of node and edge, and achieves a performance close to Traverse, which demonstrates the effectiveness of our framework. Note that Traverse asks the agent to traverse every viewpoint in the unseen scene which is an unsolved problem and always requires manual assistance in reality. While our method fully exploits the embodied capacity of the robot to explore the scene automatically and reaches a comparable performance with much fewer NoS.

2. The increase in recall rate is often accompanied by a decrease in precision. This is mainly caused by the imperfection of the Mask RCNN and Local Scene Graph Prediction Network(LSGPN) since the errors at some specific viewpoints could be hard to update and would accumulate as the agent increasingly covers the scene. Our method achieves the highest F1-score on both tasks, which illustrates that our framework handles the balance of precision and recall rate well.

3. SGGen is much more difficult than RelCls since both the label and the mask of the detecting result could be wrong, and the detecting error could further affect the generated $LSSG$ and $GSSG$. This explains the drop of all metrics in SGGen compared with those in RelCls. No matter how difficult the task is, the baseline methods only implement spatial exploration and adopt the same exploring path. While our method explores the scene at a semantic level indicating that the agent is aware of $GSSG$ and would try to update and add the semantic information of $GSSG$ in subsequent steps.

4. From table II we could discover that in RelCls, both $LSSG$ and $GSSG$ benefit the task, and the improvement of introducing $LSSG$ is more obvious. This may be because the $LSSG$ sequence encoded by LSTM contains a certain amount of $GSSG$ information. While in SGGen, adding $GSSG$ vectors is worse than Base method. We think this could be caused by the more frequently changing frequency on $GSSG$ and its vector which is very different from training. $LSSG+GSSG$ performes best on both tasks, which illustrate the effectiveness and robustness of our framework.

6.3.2 Qualitative Analysis

Fig.6 shows the representative trajectory generated by the final navigation model trained with $LSSG + GSSG$ method. The agent takes 14 steps to explore the scene. From the vertical direction of the figure we can clearly see the updating process of nodes and edges of GSSG. We can see that categories of the nodes in the generated global scene graph is rather accurate, which demonstrates that the proposed weighted voting mechanism can effectively solves the imperfection of Mask RCNN. However, there is one error in this example where the Mask RCNN mistakenly

detects an L-shaped counter top as two counter tops. However, this is not a common situation and it is hard to be fixed. The horizontal direction of Fig.6 exhibits the dynamic updating process of the global scene graph, which includes ADD, UPDATE and REPLACE operations over nodes and edges. With the local scene graphs generated at each step, the global scene graph is incrementally constructed and contains more and more semantic information.

We also find the steps where an node is added and its corresponding relationship is added are different (e.g the $lettuce$ in Fig.6). This phenomenon well illustrates the gap between detecting the object and extracting the semantic relations of the object in the scene. With the learned navigation policy, the agent first explores area where the objects are placed on the counter top and the $lettuce$ is detected at frame 2, while no relationship is detected at this time. And then the navigate policy guides the agent to explore nearby areas, and other small objects placed on the counter top can be detected. For further exploration, the policy guides the agent to move away from current viewpoint and an overall view of the counter top is captured, which corresponds to frame 7. At the same time, the relation between $lettuce$ and $counter\ top$ is finally extracted from the visual input and added to the global scene graph. In this example, we can see that the white dining table is also detected in the first several steps. With the learned policy, the agent is able to explore the nearby areas around the counter top first and then move to the dining table for further exploration. It can be seen that in frame 15, the bread, side table and shelf which are invisible from previous viewpoints are detected. After adding these newly detected objects, the agent considers that it has fully explored the scene and the stop action is triggered.

From the above examples we can find that the navigating policy would guide the agent to detect more objects and extract more semantic relationships at the same time with a relatively short path. Furthermore, the agent leverages its embodiment ability to trigger stop action when it thinks that the scene is fully explored. More results can be found in the supplemental material.

6.4 Application to Streaming Video Captioning

The generated global semantic scene graph can be further used to many downstream tasks such as captioning, question answering, and visual navigation. For more illustration of other downstream tasks, Please see the supplementary material.

Different from previous scene graph captioning work like [28], which extracts the scene graph and generates the caption for single image, our scene graph is updated in real time while the agent exploring the scene. Therefore, it is possible for us to utilize the incrementally generated scene graph for video captioning. In this work, we utilize a *scene graph guided captioning method for streaming video* to generate the captions at each step. This work generates frame captions and utilizes the rich semantic information of the generated scene graph to construct connections between frame captions and supplement the video captions. As shown in Fig.6, the video caption generated at each frame is placed on the last row. The generation process of video caption is consistent with a traverse over the global scene graph. In the video caption generated at frame 2, the sentence 'In the room there is also a lettuce, a blinds and a coffee machine' is quite confusing, since the relationships between these objects are not detected yet. However, in video caption generated at frame 7, we have the '*coffee machine* is standing on the first counter top' and the '*lettuce* is sitting on the first counter top'. The caption becomes more clear with more semantic relationship edges added into the global scene graph. At frame 15, the global scene graph becomes a dense graph, and the position of most objects could be accurately described in the caption. More examples can be found in the anonymous accompanying video which is available at https://embodiedscenegraph.vercel.app/.

7 Conclusion

In this paper, a novel embodied semantic scene graph generation framework is established and the hybrid imitation reinforcement learning method is developed to address this new task. This research paves a new path for the agent to semantically describe the environment and provides some fundamental components for some challenging down-streaming tasks such as streaming questions and answering, which has been investigated in our validation. This work can be extended for large-scale autonomous scene graph generation in complicated scenarios.

References

[1] L. Mahon, E. Giunchiglia, B. Li, and T. Lukasiewicz. Knowledge graph extraction from videos. *arXiv preprint arXiv:2007.10040*, 2020.

[2] P. Xu, X. Chang, L. Guo, P.-Y. Huang, X. Chen, and A. G. Hauptmann. A survey of scene graph: Generation and application. *EasyChair Preprint*, (3385), 2020.

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

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

[5] B. Pan, H. Cai, D.-A. Huang, K.-H. Lee, A. Gaidon, E. Adeli, and J. C. Niebles. Spatio-temporal graph for video captioning with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10870–10879, 2020.

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

[7] S. Garg, N. Sünderhauf, F. Dayoub, D. Morrison, A. Cosgun, G. Carneiro, Q. Wu, T.-J. Chin, I. Reid, S. Gould, et al. Semantics for robotic mapping, perception and interaction: A survey. *arXiv preprint arXiv:2101.00443*, 2021.

[8] U.-H. Kim, J.-M. Park, T.-J. Song, and J.-H. Kim. 3-d scene graph: A sparse and semantic representation of physical environments for intelligent agents. *IEEE transactions on cybernetics*, 50(12):4921–4933, 2019.

[9] I. Bozcan and S. Kalkan. Cosmo: Contextualized scene modeling with boltzmann machines. *Robotics and Autonomous Systems*, 113:132–148, 2019.

[10] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5664–5673, 2019.

[11] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone. 3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans. *arXiv preprint arXiv:2002.06289*, 2020.

[12] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–10, 2018.

[13] S. Tan, H. Liu, D. Guo, X. Zhang, and F. Sun. Towards embodied scene description. *RSS*, pages 1–8, 2021.

[14] Z. Fang, A. Jain, G. Sarch, A. W. Harley, and K. Fragkiadaki. Move to see better: Towards self-supervised amodal object detection. *arXiv preprint arXiv:2012.00057*, 2020.

[15] S. Tan, W. Xiang, H. Liu, D. Guo, and F. Sun. Multi-agent embodied question answering in interactive environments. *ECCV*, pages 1–8.

[16] D. S. Chaplot, H. Jiang, S. Gupta, and A. Gupta. Semantic curiosity for active visual learning. In *European Conference on Computer Vision*, pages 309–326. Springer, 2020.

[17] R. Druon, Y. Yoshiyasu, A. Kanezaki, and A. Watt. Visual object search by learning spatial context. *IEEE Robotics and Automation Letters*, 5(2):1279–1286, 2020.

[18] Z. Zeng, A. Röfer, and O. C. Jenkins. Semantic linking maps for active visual object search. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1984–1990. IEEE, 2020.

[19] A. Kurenkov, R. Martín-Martín, J. Ichnowski, K. Goldberg, and S. Savarese. Semantic and geometric modeling with neural message passing in 3d scene graphs for hierarchical mechanical search. *arXiv preprint arXiv:2012.04060*, 2020.

[20] H. Du, X. Yu, and L. Zheng. Learning object relation graph and tentative policy for visual navigation. In *European Conference on Computer Vision*, pages 19–34. Springer, 2020.

[21] F. K. Kenfack, F. A. Siddiky, F. Balint-Benczedi, and M. Beetz. Robotvqa - a scene-graph-and deep-learning-based visual question answering system for robot manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, USA*, 2020.

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

[23] J. Johnson, A. Gupta, and L. Fei-Fei. Image generation from scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1219–1228, 2018.

[24] S. Wiseman and A. M. Rush. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960*, 2016.

[25] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, D. Gordon, Y. Zhu, A. Gupta, and A. Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv*, 2017.

[26] T. Chen, S. Gupta, and A. Gupta. Learning exploration policies for navigation. *arXiv preprint arXiv:1903.01959*, 2019.

[27] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97.'Towards New Computational Principles for Robotics and Automation'*, pages 146–151. IEEE, 1997.

[28] X. Yang, K. Tang, H. Zhang, and J. Cai. Auto-encoding scene graphs for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10685–10694, 2019.