Learning Hierarchical Graph Convolutional Neural Network for Object Navigation

Tao Xu[1,2], Xu Yang[1,2], and Suiwu Zheng[1,2,3](✉)

[1] State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China
{xutao2020,xu.yang,suiwu.zheng}@ia.ac.cn

[2] University of Chinese Academy of Sciences, Beijing, China

[3] Huizhou Zhongke Advanced Manufacturing Limited Company, Huizhou, China

Abstract. The goal of object navigation is to navigate an agent to a target object using visual input. Without GPS and the map, one challenge of this task is how to locate the target object in the unseen environment, especially when the target object is not in the field of view. Previous works use relation graphs to encode the concurrence relationships among all the object categories, but these relation graphs are usually too flat for the agent to locate the target object efficiently. In this paper, a Hierarchical Graph Convolutional Neural Network (HGCNN) is proposed to encode the object relationships in a hierarchical manner. Specifically, the HGCNN consists of two graph convolution blocks and a graph pooling block, which constructs the hierarchical relation graph by learning an area-level graph from the object-level graph. Consequently, the HGCNN based framework enables the agent to locate the target object efficiently in the unseen environment. The proposed model is evaluated in the AI2-iTHOR environment, and the performance of object navigation shows a significant improvement.

Keywords: Object navigation · Hierarchical relation graph · AI2-iTHOR

1 Introduction

Object navigation [2,5,18] provides a fundamental capability for robots and other unmanned systems, which has attracted great attention in the past few years with the development of various embodied AI simulators.

The goal of object navigation is to navigate an agent to a specified target using the first-view visual input such as RGB image. To achieve this goal, the agent should learn about visual representation and navigation policy. For instance, when an agent is asked to find an apple in an unseen environment, it should know the appearance of the apple and how to search it in the environment. Furthermore, the agent should also be able to reason, especially in the unseen environment, because targets are often out of sight at the first location.

E. Pimenidis et al. (Eds.): ICANN 2022, LNCS 13530, pp. 544–556, 2022.
https://doi.org/10.1007/978-3-031-15931-2_45

Recently, to learn an informative visual representation and gain the ability of reasoning, Graph Convolutional Network (GCN) has been introduced into this task to construct relation graphs [5,11] among the object categories in the environment. However, almost all existing relation graphs used in this task are inherently flat, which means that they can only encode the local object-level relationships among the object categories and ignore the hierarchical structure of the relation graphs. On the other hand, the hierarchical architecture of knowledge is key for humans to learn how to navigate to a target. For example, when a person is asked to find a plunger in a house, he may first recall that the plunger, toilet paper and toilet always appear in the same area. Then he will search for that area in the house because it is easier to find the area than the plunger in the big house. Finally, when he gets to the area, he will search for the plunger in the area and find it eventually. The example shows that the hierarchical architecture of knowledge makes human's decision-making more efficient, which inspires us to design the agent acting like that.

The main challenge is how to learn the hierarchical architecture of the relationships among the object categories in the environment. Specifically, the model needs to extract the area-level concepts from the existing object categories and encode the relationships among them in the area-level relation graph. Learning how to utilize the hierarchical relation graph to navigate is another challenge. We notice that Zhang *et al.* [22] implement a Hierarchical Object-to-Zone (HOZ) graph to learn hierarchical relationships among the object categories, which makes a significant improvement over the baseline. However, the HOZ graph is limited by setting the so-called current zone, sub-goal zone and target zone by rules, for the agent to get an efficient navigation policy. And the HOZ graph uses a heuristic method to generate the area-level graph rather than a learning scheme.

In this paper, a novel Hierarchical Graph Convolutional Neural Network (HGCNN) is proposed to encode the hierarchical relationships among the object categories for navigation. Specifically, the HGCNN consists of two graph convolution blocks and a graph pooling block, where the graph convolution blocks encode the relationships among the nodes, and the graph pooling block provides a way to cluster the nodes in the object-level relation graph together and then extract the area-level concepts as new nodes. After that, an area-level relation graph with the new nodes can be constructed. Besides, an LSTM layer is implemented as the navigation policy module to learn to utilize the hierarchical relation graph to navigate in the unseen environment. By end-to-end reinforcement learning, the agent is able to utilize the hierarchical object relationships to reason and navigate to the target efficiently.

The proposed model is evaluated in the AI2-iTHOR [9] environment, and the experiment shows a significant improvement over the baseline. The main contributions of this paper can be summarized as follows:

– A novel Hierarchical Graph Convolutional Neural Network (HGCNN) is proposed to encode the hierarchical relation graph for object navigation.

– This paper modifies the DIFFPOOL [21] layer and introduces it into the HGCNN, which facilitates the task a lot.
– With the implementation of the HGCNN, the performance of object navigation shows a significant improvement.

2 Related Work

Visual Navigation. According to the different ways of setting goals, target-driven visual navigation can be divided into ImageGoal navigation [11,23], PointGoal navigation [17] and object navigation. This paper focuses on object navigation, which aims to navigate an agent to a target object in the unseen environment. Conventional works usually treat visual navigation as a geometric problem. Oriolo *et al.* [14] use a given map to navigate the robots. Gupta *et al.* [6] divide it into two parts: mapping and planning, which makes it more interpretable. However, these approaches usually have problems in dealing with complex environments or situations. So reinforcement learning (RL, *e.g.*, A3C [13] and DDPPO [17]) is introduced into this task to learn the complex policy [23]. However, instead of relying on the map or focusing on the training algorithm, this paper proposes the HGCNN to learn an visual representation to provide more prior knowledge for navigation.

Object Navigation. Recent works improve generalization to unknown environment with different strategies. Chaplot *et al.* [2] build an episodic semantic map to explore the environment. Maksymets *et al.* [12] propose a data augmentation technique to address overfitting in object navigation. Ye *et al.* [20] add auxiliary learning tasks to improve the generaliazation to unknown environment. This line of works have achieved great results but they also rely on other inputs besides RGB image, such as depth image and GPS coordinate. In this paper, we propose a novel method to navigate the agent to the target object only using the RGB input.

Learning Visual Representation Based on Graph. Learning visual representation aims to extract an informative feature vector from the visual input. Moreover, many works use graphs to encode prior knowledge to facilitate visual navigation. Du *et al.* [5] introduce an Object Relation Graph (ORG) which can be learned during end-to-end training to encode the concurrence relationships among object categories. Yang *et al.* [19] leverage a knowledge graph to encode the semantic scene priors. Lv *et al.* [11] use a relation graph to encode the 3D spatial relationships among object categories. Taniguchi *et al.* [16] propose a graph-based memory to encode the topological memory for navigation. However, the graphs this line of works construct are mostly flat, which means that the visual representation they learn can only encode the local feature of the environment. In this paper, the HGCNN is proposed to construct the hierarchical relation graph, which contains a subset of global object relationships.

Graph Pooling. The goal of graph pooling is to extract the global feature from the original graph. In particular, there are three different mechanisms for graph

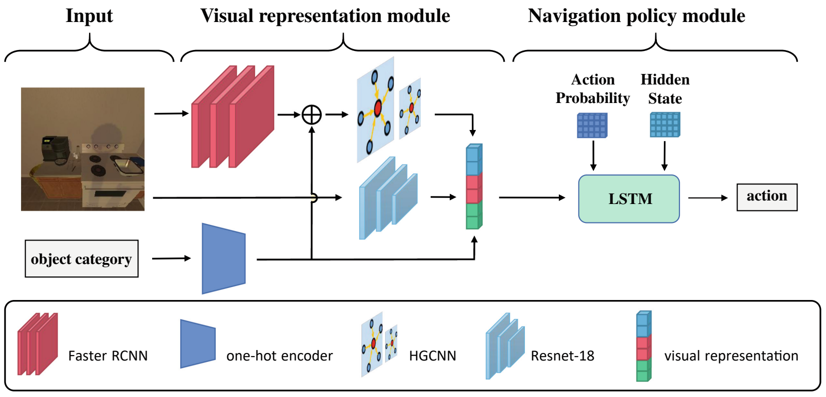

Fig. 1. Model overview. The proposed model includes two main modules: the visual representation module and the navigation policy module. The visual representation module takes the current first-view image and the target object category as input and generates a visual representation. Given the visual representation, previous action probability and hidden state, the navigation policy module outputs the current action probability.

pooling, and they are mechanisms based on graph coarsening [21], mechanisms based on edge contraction [4] and mechanisms based on TopK [10] respectively. In this paper, the HGCNN focuses on mechanisms based on graph coarsening (*e.g.*, DIFFPOOL [21]). Different from the conventional graph pooling, in this paper, the standard DIFFPOOL is modified according to the task specificity and introduced into the HGCNN, which facilitates the task a lot.

3 Model

In this section, the task definition of object navigation is introduced at the beginning. Then this paper provides an overview of the proposed model. Finally, the HGCNN and how it works to construct the hierarchical relation graph are expatiated.

3.1 Task Definition

Given an object category, the goal of the task in this paper is to navigate an embodied agent to a specified target only using the first-view visual input such as RGB image, while the conventional navigation input is not available such as GPS. At the beginning of each episode, the agent is initialized at a random state $S = \{x, y, \theta_r, \theta_h\}$ in a random scene, where x and y represent the location of the agent, θ_r and θ_h represent the point of view of the agent. With the first-view visual input, the agent should select a series of actions from $\Lambda = \{MoveAhead, RotateLeft, RotateRight, LookDown, LookUp, Done\}$ to navigate to the object which belongs to the given object category. At the end of each episode, when the target object is in the field of view and the distance between the agent and the target object is less than a threshold(*e.g.*, 1.5 m) and the agent chooses the action *Done*, the experiment counts this episode as a successful episode, otherwise it is a failure one.

3.2 Model Overview

The proposed model has two main modules: the visual representation module and the navigation policy module (Fig. 1). Specifically, the visual representation module has three main components: an HGCNN, a Resnet-18 [7] and an object category encoder. The navigation policy module is an LSTM layer to learn how to utilize the visual representation to predict efficient actions.

First, given an object category and a current first-view image, the proposed model employs a Faster RCNN [15] to get the local detection feature which consists of the bounding box positions and detection confidence. After that, the model concatenates it with the object embedding as location-aware feature (LAF) following [5]. And then the HGCNN encodes the hierarchical relationships among the object categories, the Resnet-18 which is pretrained on ImageNet [3] embeds the current first-view image to extract the visual feature, and the object category encoder uses a one-hot vector to encode the target object category. After that, three new embeddings are generated and they are concatenated as a visual representation vector. Finally, the model inputs the visual representation vector, the previous action and the hidden state to the LSTM layer to predict efficient actions.

3.3 HGCNN

Similar to Convolutional Neural Network (CNN), the proposed HGCNN consists of two main components: the graph convolution block and the graph pooling block (Fig. 2).

Graph Convolution Block. The goal of the graph convolution block is to learn the object representation embedding. Consider such a graph $G = (N, A)$, each node $n \in N$ denotes the LAF extracted from the current first-view image, and each edge $a \in A$ denotes the relationship among different nodes. The graph convolution block uses a multi-layer GCN to learn the object representation embedding, and a GCN layer is expressed as:

$$Z = f(AXW_{conv}) \tag{1}$$

where A denotes the adjacent matrix, X denotes all the nodes, W_{conv} denotes the embedding matrix and $f(\cdot)$ denotes the ReLU activation function. After that, an object representation embedding Z is generated.

Graph Pooling Block. The graph pooling block provides a way to cluster the nodes in the object-level graph together and then extract the area-level concepts as new nodes (Fig. 3). After that, an area-level graph can be constructed. In

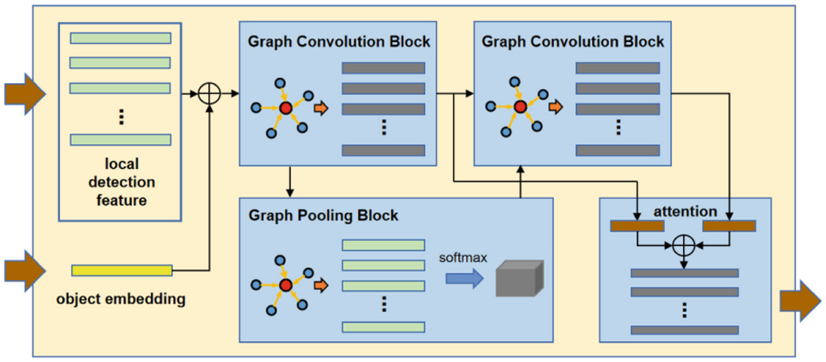

Fig. 2. Detailed structure of the HGCNN. The HGCNN consists of two main components: the graph convolution block and the graph pooling block. The graph convolution block aims to learn the object representation embedding. And the goal of the graph pooling block is to cluster the nodes in the object-level relation graph to extract the area-level concepts as new nodes and then construct the area-level relation graph. Additionally, at the end of the HGCNN, a graph attention layer is introduced to obtain the final feature.

the graph pooling block, a modified DIFFPOOL [21] layer is implemented to achieve this goal. Specifically, the graph pooling block first uses a multi-layer GCN to learn a cluster assignment matrix S from the object-level graph and then constructs a new area-level graph using the cluster assignment matrix S. The number of columns of S is a hyperparameter that corresponds to the number of areas. In particular, the following equations are applied:

$$S = \mathrm{softmax}(f(AXW_{pool})) \tag{2}$$

$$X' = S^T Z \tag{3}$$

$$A' = S^T A S^T \tag{4}$$

where W_{pool} denotes the pooling embedding matrix, S denotes the assignment matrix, X' denotes all the nodes in the area-level graph, and A' denotes the new adjacent matrix. After that, a new coarsened area-level graph can be constructed. Then the HGCNN uses the other graph convolution block to learn an area-level object representation embedding.

In the standard DIFFPOOL [21] layer, the cluster assignment matrix S provides an assignment of each node at the low layer to the new cluster at the high layer. Each row of S corresponds to a node at the low layer, and each column of S corresponds to a cluster at the high layer. Therefore, each row of S should be close to a one-hot vector because each node at the low layer can only cluster to one node at the high layer. However, in this work, the distribution of objects in the environment always changes, which means that an object not always clusters to an area. In other words, each row of S does not need to be close to a one-hot vector. So this constraint is removed in the model.

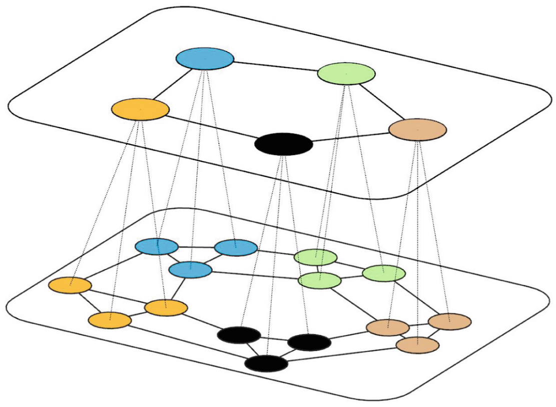

Fig. 3. Graph pooling block. The goal of the graph pooling block is to cluster the nodes in the object-level relation graph to extract the area-level concepts as new nodes and then construct the area-level relation graph.

Model Details. In the HGCNN, the first graph convolution block uses two GCN layers to learn the object-level representation embedding. Then the graph pooling block uses two GCN layers to learn the assignment matrix S. Finally, the second graph convolution block uses another two GCN layers to learn the area-level representation embedding.

After the graph convolution blocks, two new embeddings Z_1 and Z_2 are generated. Following [5], a graph attention layer is introduced to obtain the attended location-aware appearance feature. By concatenating it with LAF, the HGCNN outputs the final feature.

4 Experiments

4.1 Environment Settings

The proposed model is evaluated on AI2-iTHOR v1.0.1, which contains four types of scenes, and each of them contains 30 rooms. Following [18], a subset of 22 object categories are selected from the four types of scenes to ensure that there are more than 4 objects in each scene. Moreover, the rooms from each scene are divided into three parts, 20 as the training set, 5 as the validation set, and the remaining 5 as the testing set.

4.2 Evaluation

In this paper, the experiment uses success rate (SR) and success weighted by path length (SPL) [1] to evaluate the proposed methods. SR is the success rate of the agent finding the target objects, which is defined as the ratio of the number of successful episodes and the number of all the episodes. SPL aims to evaluate the proximity between the actual path and the optimal path, which is formulated as $SPL = \frac{1}{N}\Sigma_{i=1}^{N}\delta(i)\frac{L_i^{opt}}{\max(L_i, L_i^{opt})}$, where N denotes the number of episodes, L_i^{opt} and L_i represent the optimal path length and the actual path length in episode i respectively, and $\delta(i)$ is a binary indicator to indicate whether the episode i is successful. We report results for all targets (ALL) and a subset of targets ($L \geq 5$) whose length of optimal path is no less than 5.

4.3 Implementation Details

In this paper, a fundamental model which simply inputs the feature extracted by the Faster RCNN to the LSTM layer to predict the actions is chosen as the baseline. The A3C algorithm is introduced to train the proposed model for 6M episodes with 12 asynchronous workers. In each episode, the agent is penalized every step with -0.01, which forces the agent to learn the most efficient policy. At the end of each episode, the agent will get a reward of 5 if the episode is successful. The Faster RCNN in the model is pretrained and finetuned on AI2-iTHOR dataset. To update the network, the experiment employs Adam optimizer [8] with a learning rate of 10^{-4}.

Besides, all the models are evaluated in the validation set for 250 episodes in each scene type, and the model with the highest success rate is selected. Due to the stochasticity of the inference, the experiment performs each model 5 times in the testing set and calculates the average of all metrics as the final result.

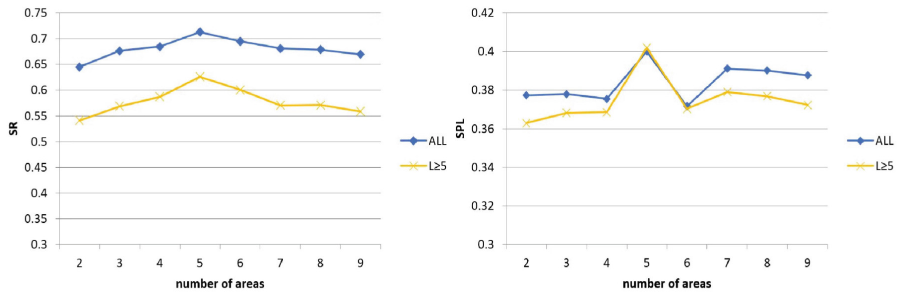

Fig. 4. Impact of the number of areas.

4.4 Ablation Study

Impact of the Number of Areas. In this paper, the number of objects is set to 22, and the number of areas is a hyperparameter that determines the size of the area-level relation graph. In theory, whether the number of areas is too large or too small, the HGCNN will be less effective, which can be proved in Fig. 4. As indicated in Fig. 4, the optimal number of areas in the experiment is 5. So the number of areas in the remaining experiments is set to 5.

Table 1. Comparisons of different graph pooling block.

Method	ALL		$L \geq 5$	
	SR	SPL	SR	SPL
NO pooling	0.663	0.377	0.564	0.368
Standard DIFFPOOL	0.678	0.386	0.587	0.381
Modified DIFFPOOL	0.713	0.400	0.626	0.402

Impact of the Graph Pooling Block. In the proposed model, the HGCNN uses the graph pooling block to cluster the nodes and extract the area-level concepts as new nodes. According to the task specificity, the DIFFPOOL layer is modified and introduced into the HGCNN. Table 1 indicates that the model with modified DIFFPOOL layer outperforms the model with standard DIFF-POOL layer and the model without graph pooling block. So in the remaining experiments, the modified DIFFPOOL layer is introduced to cluster the nodes and extract the area-level nodes.

Table 2. Comparisons of navigation results.

Method	ALL		$L \geq 5$	
	SR	SPL	SR	SPL
baseline	0.618	0.345	0.494	0.316
baseline+ORG [5]	0.663	0.377	0.564	0.368
baseline+HGCNN (ours)	0.713	0.400	0.626	0.402

4.5 Quantitative Results

The quantitative results are shown in Table 2. As seen in Table 2, the proposed model outperforms the baseline and the baseline with ORG in all four metrics, which justifies the effect of the HGCNN. The baseline simply inputs the detection results to the navigation policy module, which means that it can not utilize the object relationships to reason and locate the target object. Moreover, the baseline with ORG uses an object-level graph to encode the concurrence relationships, which ignores the hierarchical architecture of relation graph. Compared with them, the model with HGCNN encodes the hierarchical relationships among object categories, which is more efficient for the agent to locate the target object. Specifically, as shown in Fig. 5, when the environment is complex and the target

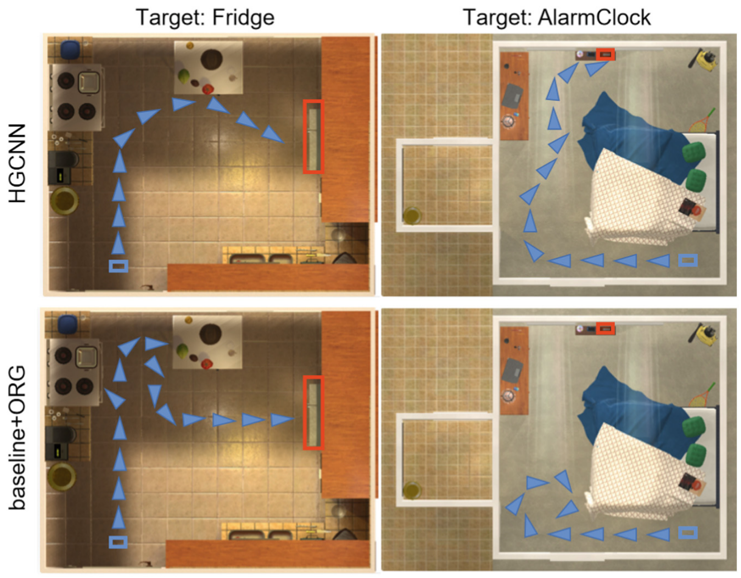

Fig. 5. Visual results of different models. We compare the HGCNN model with the baseline with ORG. The target objects are highlighted by red bounding boxes and the blue squares represent the initial position of agent.

object is not in the field of view, the baseline with ORG often takes more steps or even fails to reach the target object while the HGCNN model can always provide efficient policy based on the hierarchical relation graph.

Besides, this paper also compares the proposed model with the HOZ graph and shows the results in Table 3. As indicated in Table 3, the proposed model achieves better results especially in SPL($L \geq 5$) metric. Moreover, the experiment shows that the HOZ graph uses average 36.5 steps to reach the target object while the HGCNN only uses average 17.1 steps to achieve it, which means that compared to the HOZ graph, the HGCNN model provides more efficient navigation policy rather than rotating on the spot. It may be caused by the rules the HOZ graph uses to set the current zone, sub-goal zone and target zone during navigation, which is too limited for the agent to learn an efficient policy. Moreover, the HOZ graph uses a heuristic method to generate the area-level graph while the HGCNN uses a learnable graph pooling block to achieve it.

Table 3. Comparison with the HOZ graph.

Method	ALL		$L \geq 5$	
	SR	SPL	SR	SPL
baseline+HOZ [22]	0.706	0.400	0.628	0.392
baseline+HGCNN (ours)	0.713	0.400	0.626	0.402

Furthermore, in theory, when the environment is complex and the number of object categories is too large, it is difficult for the HOZ graph to extend the layers of the hierarchical graph to enhance the effect. However, the proposed HGCNN can simply stack the graph convolution blocks and graph pooling blocks to construct a multi-layer hierarchical relation graph to facilitate the task.

5 Conclusion

In this paper, a Hierarchical Graph Convolutional Neural Network (HGCNN) is proposed to learn an informative visual representation for object navigation. With the HGCNN, the agent can locate the target object more efficiently in the unseen environment. Moreover, a DIFFPOOL layer is modified according to the task specificity and introduced into the HGCNN, which facilitates the task a lot. The experiment shows a significant improvement over the baseline. In future work, fusing the features extracted from different graph layers better and applying the model to more complex environments are two areas of concern.

Acknowledgement. This work was supported by the National Key Research and Development Program of China under Grant 2020AAA0105900, partly by National Natural Science Foundation (NSFC) of China (grants 91948303, 61973301, 61972020), partly by Youth Innovation Promotion Association CAS, and partly by Beijing Science and Technology Plan Project (grant Z201100008320029).

References

1. Anderson, P., et al.: On evaluation of embodied navigation agents. arXiv preprint arXiv:1807.06757 (2018)
2. Chaplot, D.S., Gandhi, D.P., Gupta, A., Salakhutdinov, R.R.: Object goal navigation using goal-oriented semantic exploration. In: Advances in Neural Information Processing Systems, pp. 4247–4258 (2020)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009)
4. Diehl, F.: Edge contraction pooling for graph neural networks. arXiv preprint arXiv:1905.10990 (2019)
5. Du, H., Yu, X., Zheng, L.: Learning object relation graph and tentative policy for visual navigation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) ECCV 2020. LNCS, vol. 12352, pp. 19–34. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58571-6_2

6. Gupta, S., Davidson, J., Levine, S., Sukthankar, R., Malik, J.: Cognitive mapping and planning for visual navigation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2616–2625 (2017)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
8. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: International Conference on Learning Representations (Poster), pp. 1–15 (2015)
9. Kolve, E., et al.: Ai2-thor: An interactive 3d environment for visual ai. arXiv preprint arXiv:1712.05474 (2017)
10. Lee, J., Lee, I., Kang, J.: Self-attention graph pooling. In: International Conference on Machine Learning, pp. 3734–3743 (2019)
11. Lv, Y., Xie, N., Shi, Y., Wang, Z., Shen, H.T.: Improving target-driven visual navigation with attention on 3d spatial relationships. arXiv preprint arXiv:2005.02153 (2020)
12. Maksymets, O., et al.: Thda: treasure hunt data augmentation for semantic navigation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 15374–15383 (2021)
13. Mnih, V., et al.: Asynchronous methods for deep reinforcement learning. In: International Conference on Machine Learning, pp. 1928–1937 (2016)
14. Oriolo, G., Vendittelli, M., Ulivi, G.: On-line map building and navigation for autonomous mobile robots. In: Proceedings of 1995 IEEE International Conference on Robotics and Automation, pp. 2900–2906 (1995)
15. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
16. Taniguchi, A., Sasaki, F., Yamashina, R.: Pose invariant topological memory for visual navigation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 15384–15393 (2021)
17. Wijmans, E., et al.: DD-PPO: learning near-perfect pointgoal navigators from 2.5 billion frames. In: International Conference on Learning Representations, pp. 1–21 (2019)
18. Wortsman, M., Ehsani, K., Rastegari, M., Farhadi, A., Mottaghi, R.: Learning to learn how to learn: Self-adaptive visual navigation using meta-learning. In: Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition, pp. 6750–6759 (2019)
19. Yang, W., Wang, X., Farhadi, A., Gupta, A., Mottaghi, R.: Visual semantic navigation using scene priors. In: International Conference on Learning Representations, pp. 1–13 (2019)
20. Ye, J., Batra, D., Das, A., Wijmans, E.: Auxiliary tasks and exploration enable objectgoal navigation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 16117–16126 (2021)
21. Ying, R., You, J., Morris, C., Ren, X., Hamilton, W.L., Leskovec, J.: Hierarchical graph representation learning with differentiable pooling. In: Advances in Neural Information Processing Systems, pp. 4805–4815 (2018)

22. Zhang, S., Song, X., Bai, Y., Li, W., Chu, Y., Jiang, S.: Hierarchical object-to-zone graph for object navigation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 15130–15140 (2021)
23. Zhu, Y., Mottaghi, R., Kolve, E., Lim, J.J., Gupta, A., Fei-Fei, L., Farhadi, A.: Target-driven visual navigation in indoor scenes using deep reinforcement learning. In: Proceedings of the IEEE International Conference on Robotics and Automation, pp. 3357–3364 (2017)