Self-Supervised Multi-Granularity Map Learning for Vision-and-Language Navigation

Anonymous Author(s)
Affiliation
Address
email

Abstract

We address a practical yet challenging problem of training robot agents to navigate in an environment following a path described by some language instructions. The instructions often contain descriptions of objects in the environment and path cues defined by humans. To achieve accurate and efficient navigation, it is critical to build a map that accurately represents both spatial location and the semantic information of the environment objects. However, enabling a robot to build a map that well represents the environment is extremely challenging as the environment often involves diverse objects with various attributes. In this paper, we propose a multi-granularity map, which contains both object fine-grained details (*e.g.*, color, texture) and semantic classes, to represent objects more comprehensively. Moreover, we propose a self-supervised auxiliary task, which requires the agent to localize instruction-relevant objects on the map. Through this task, the agent not only learns to localize the instruction-relevant objects for navigation but also is encouraged to learn a better map representation that reveals object information. We then feed the learned map and instruction to a waypoint predictor to determine the next navigation goal. Experimental results show our method outperforms the state-of-the-art by 4.0% and 4.6% *w.r.t.* success rate both in seen and unseen environments, respectively on VLN-CE dataset.

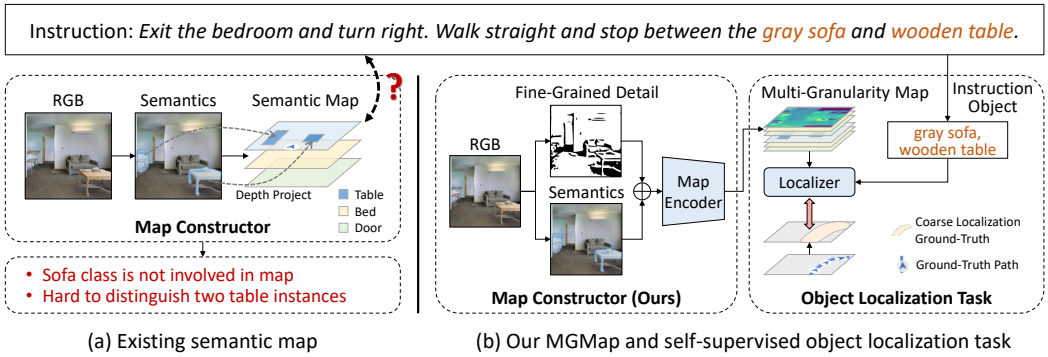
1 Introduction

Developing a robot that is able to cooperate with humans is one of the goals for embodied artificial intelligence. An important ability for such a robot is to understand human instructions (*e.g.*, "*walk to gray sofa and stop at wooden table*") and navigate to the corresponding location. Toward this goal, Anderson *et al.* [2] propose a vision-and-language navigation (VLN) task, which requires an agent to follow human language instructions to navigate in unseen environments.

Current dominant methods for this task are designed in an end-to-end manner [29, 36, 44]. They perceive the environment implicitly from raw RGB-D images and predict a sequence of actions using a recurrent model. These methods attempt to learn mapping, instruction-vision correspondence, and path planning implicitly, which increases the learning difficulty. In contrast, we humans will build a map explicitly to represent environments [41, 49]. The map provides necessary information for locating objects that are mentioned in the instruction and finding the corresponding path.

Enabling an embodied agent to mimic humans, learning a map that represents environment comprehensively, is a challenging problem, because the environment involves diverse object classes and each object is with various attributes (*e.g.*, *long wooden table*). Previous methods [10, 22, 30] try to build a semantic map for the agent through projecting RGB image segmentation results to an egocentric top-down map as shown in Figure 1 (a). This map reveals the location and semantic class of a part of environment objects. However, it is hard to represent all object classes (*e.g.*, sofa is not repre-

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.



(a) Existing semantic map (b) Our MGMap and self-supervised object localization task

Figure 1: Existing semantic map (a) can only represent a part of environment object classes without attribute details. Our multi-granularity map (b) contains extra fine-grained environment details (*e.g.*, texture, color) and learns to represent diverse objects with detailed attributes through a self-supervised object localization task.

sented). More critically, it can not represent the attributes of objects (*e.g.*, shape, color, texture and material), which are critical for localizing an object. As a result, it provides insufficient information to discriminate the "*long wooden dining table*" from two table instances shown in Figure 1 (a). One naive solution is to annotate a dataset covering a large number of object classes and attributes for training a perfect segmentation model. This solution is labor-intensive. It is also infeasible to build a class list covering all possible object classes and attributes appearing in the complex environment.

In this paper, we propose to learn a map that represents semantics and spatial location of diverse objects in a self-supervised manner. We achieve this goal in two steps. First, we augment existing semantic map to a multi-granularity map as shown in Figure 1 (b). This map contains different granularity information, namely high-level object semantics which is recognized by a segmentation model, and low-level object fine-grained details. The fine-grained details are extracted from segmentation model latent features which are proven to contain rich object details such as color, texture, and shape [31, 55].

Second, to make the map better represent instruction objects using the multi-granularity features described above, we propose an instruction-relevant object localization auxiliary task as shown in Figure 1 (b). Specifically, we feed the map representation and instruction objects to a localizer to predict the location of these objects. Instead of manually annotating the object location on a map as localization ground-truth, we automatically generate a coarse localization ground-truth from instruction-path paired data, considering map regions that are close to instruction path as coarse ground-truth. From this task, agents learn to localize instruction-relevant objects for finding an instruction-relevant path. More critically, to localize instruction objects, the map encoder is encouraged to reason a map representation that reveals precise semantics of each map region.

We then feed the learned map and the instruction to a waypoint predictor to determine the next navigation goal. An existing off-the-shelf local policy [52] is used to determine a low-level action (*e.g.*, go forward, turn left or right) to go for the navigation goal. We name our method as self-supervised multi-granularity map (**SS-MGMap**) for VLN. Experimental results on VLN-CE benchmark dataset show our proposed method outperforms state-of-the-art methods. To sum up, our main contributions are as followings:

- We propose a multi-granularity map to represent both fine-grained details and abstract semantics information of environment. To our best knowledge, it is the first time to introduce multi-granularity knowledge in a map format for VLN task.
- We propose a self-supervised object localization auxiliary task, from which agents learn to leverage multi-granularity information to infer a discriminative map representation without the need of manual map annotation.
- With SS-MGMap, our agent robustly localizes objects that are incorrectly recognized by segmentation models. Experimental results on VLN-CE [36] show our method improves navigation success rate by 4.0% and 4.6% in seen and unseen environments, respectively.

2 Related Works

2.1 Vision-and-language navigation

VLN task [3] has drawn significant attention in embodied AI domain. Early works focus on the discrete environments where the agents can only perceive from a sparse set of points, including data augmentation methods [20, 48], auxiliary tasks methods [38, 51, 56] and pretraining methods [24, 25, 39]. However, these methods can not perform well when the agent navigates in continuous 3D environments [28]. To address this issue, [36] exploits Habitat simulator [46] to convert discrete trajectory paths to continuous trajectories and proposes a continuous VLN setting. Under these settings, [36, 44] propose the end-to-end method in which the agent takes a representation of the visual observation and instructions as input at each time step to predict an action. Another line of works [22, 35] tackles the VLN problems by leveraging the language-conditioned waypoint. [12, 22, 30] build a structured semantic top-down map, which is more relevant to our work. However, those map-based methods are limited in representing all object classes or the attributes of objects, which disables the agent to ground the instructions in maps very well. In contrast to all these works, our method learns a map that represents the semantics and spatial location of diverse objects in a self-supervised manner.

2.2 Map representation for navigation

Perceiving environment through a map representation is helpful for a variety of navigation tasks. Previous works propose different methods for building a map, each of them focusing on representing different information on the map. For example, occupancy map indicates whether a point is occupied and provides location information for point-goal navigation and exploration [32, 33, 43]. Topological maps encode the relationship information among different nodes in the environment and have been explored to tackle different navigation tasks [6, 11, 12, 40]. Semantic map represents the semantic information of the environment and has been successfully applied in object-goal navigation [10, 16, 54]. Concurrent works [22, 30] attempt to leverage semantic map for VLN task. However, as compared to object-goal navigation, the instructions in VLN involve more object information. It is challenging to ground rich object information in semantic map that presents a limited number of object categories. Unlike the methods described above, we propose a multi-granularity map which contains complementary granularity information for representing environment more comprehensively.

2.3 Self-supervised learning

Self-supervised learning has been widely employed in learning generic representations for images [14, 15, 19, 23, 27], video [7, 13, 37, 53], as well as multi-modal data [1, 4, 5, 34]. Recently, [24, 26, 29] propose to learn correlations between vision and text using a BERT-like [18] self-supervised objectives, and leverage the learned knowledge for VLN tasks. In contrast to learning generic representations for images or videos, we aim to learn a representative map representation that provides environment information for VLN.

3 Vision-and-Language Navigation using Multi-Granularity Map

3.1 Problem formulation

We consider the vision-and-language navigation task in a continued environment [36] (VLN-CE), where an agent is required to follow a specific path described by the natural language instruction I. Compared with the original VLN task, in VLN-CE settings, agents can not access to predefined navigable graph, so the agent is required to perform navigation through a sequence of low-level actions $a \in \mathcal{A} = \{\text{FORWARD}, \text{TURN-LEFT}, \text{TURN-RIGHT} \text{ and } \text{STOP}\}$. We equip the agent with an RGB-D camera with limited field of view, capturing a 224×224 RGB image R and depth image D at each time step. Although some current works use predefined navigable graph [12], continuous-space actions [35] and panoramic cameras [20, 28] to improve the performance, we strictly keep the same settings with VLN-CE [36], which are closer to reality and more challenging.

Existing VLN methods [35, 36, 44] attempt to perceive environment implicitly from RGB-D observation, which often requires a large number of training data. In contrast, humans will build a map-like

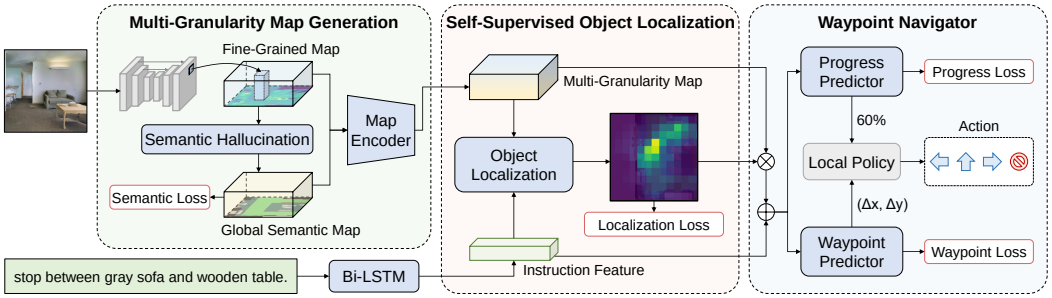


Figure 2: General scheme of SS-MGMap for VLN task. We assemble both fine-grained details and semantic information about environments to build a multi-granularity map. Agents learn to leverage such information for representing diverse objects through a self-supervised object localization task. The learned map and instruction are then fed to a waypoint navigator for deciding actions.

environment representation explicitly to provide necessary information for VLN. Equipping an agent with the mapping ability like humans is challenging because it is hard to represent diverse environment objects and their attributes in a map. In this paper, we solve this challenge from two aspects: 1) we augment a semantic map with extra fine-grained environment details (*e.g.*, texture, color), which are extracted from latent features of a pre-trained segmentation model, to build a multi-granularity map. These different granularity features provide necessary information for representing diverse objects and attributes. 2) To make the multi-granularity map representation more discriminative, we propose a self-supervised localization auxiliary task for learning correspondence between map representation and instruction objects. With the learned multi-granularity map, we feed it and the instruction to a waypoint navigator to decide the next navigation action. The general scheme is shown in Figure 2.

3.2 Perceiving environment via multi-granularity map

To better represent environment objects on a map, we propose to leverage both fine-grained features and semantic features about objects for building a multi-granularity map. We next introduce how to capture these two types of features for map representation.

Fine-grained map. To capture fine-grained environment details, we feed RGB images captured at each time step to a segmentation pre-trained model (a U-Net [45] in our case). Previous network interpretability works [31, 55] show that latent features from different layers of this model contain different types of fine-grained details of objects, *i.e.*, low-level features represent color and texture while high-level features represent object parts. Based on this observation, we use these latent features to represent environment details and project the latent features to an egocentric top-down map [8]. We name the projected map as fine-grained map $\mathbf{M}^f \in \mathbb{R}^{m \times m \times c_f}$, where $m \times m$ is map size and $c_f = 64$ is the number of channels of the projected latent features. Each map pixel represents $12cm \times 12cm$ space of environments. We choose to project latent features from the last layer of U-Net, which contains both low-level and high-level features because of skip connection mechanism in U-Net. We also evaluate the effect of other layer features in experiments.

Global semantic map. The fine-grained map aggregates environment details from RGB images which are captured along the navigation path. Because camera is with a limited field of view, the fine-grained map does not contain environmental information that has not been observed. More importantly, requiring agents to recognize all objects, including some common-seen objects, from fine-grained details increases learning difficulty. Thus, to hallucinate environments beyond the field of view and to get semantic information about common-seen objects, we feed the fine-grained map to a semantic hallucination module to predict a global semantic map $\mathbf{M}^s \in \mathbb{R}^{m \times m \times c_s}$, where each pixel indicates whether a particular object out of $c_s = 27$ object categories locates in the corresponding space. The c_s object categories are listed in Appendix. The hallucination module consists of three convolution layers with batch norm and ReLU activations, followed by a U-Net which has two encoder and two decoder convolutional blocks with skip connections. We get the ground-truth global semantic map $\bar{\mathbf{M}}^s$ from the available 3D semantic information in Matterport3D following existing methods [21, 22]. A semantic loss l_s is defined as follows:

$$l_s = \text{CrossEntropy}(\mathbf{M}^s, \bar{\mathbf{M}}^s). \tag{1}$$

Multi-granularity map. We merge information from both the fine-grained map and predicted semantic map using a map encoder. Specifically, these two maps are first processed by a convolution layer, respectively. The concatenation of two processed maps is fed to a convolution layer for procuding a multi-granularity map $\mathbf{M} \in \mathbb{R}^{m \times m \times c}$, where c is the number of map channels.

3.3 Self-supervised map representation learning via object localization task

To learn the correspondence between multi-granularity map representation and diverse objects, we propose a self-supervised instruction-relevant object localization auxiliary task. To finish this task, the agent is required to reason precise semantics information for each region of map representation, which makes the map representation more significant.

Formulation of self-supervised auxiliary task. To localize the instruction object on a map, we feed the instruction and multi-granularity map to an object localization module to predict a grid map $\hat{\mathbf{P}} \in \mathbb{R}^{m \times m}$ which indicates a 2D distribution of potential locations of instruction-relevant objects. Because we do not know the exact location of these instruction objects, we use the instruction-relevant path as guidance to generate a coarse ground-truth grid map $\mathbf{P} \in \mathbb{R}^{m \times m}$. The principle for generating coarse ground-truth is that the regions closer to the path have a higher probability of containing instruction-relevant objects. Specifically, we first calculate euclidean distance d_k from the k-th map region to the path. A normalized distance map is defined as $\mathbf{P}' = \{\frac{d_{\max} - d_k}{d_{\max} - d_{\min}}\}_{k=1}^{m \times m}$, where $d_{\max}$ and $d_{\min}$ are the maximum and minimum distances among $m \times m$ regions, respectively. For a region traversed by the path, $d_k = d_{\min} = 0$ and $\mathbf{P}'_k = 1$; for the farthest region, $d_k = d_{\max}$ and $\mathbf{P}'_k = 0$. The coarse ground-truth grid map $\mathbf{P}$ is the softmax across all grids in the distance map, *i.e.*, $\mathbf{P} = \mathrm{Softmax}(\mathbf{P}')$. An object localization loss l_o is defined as KL divergence between predicted and coarse ground-truth distributions, *i.e.*,

$$l_o = \sum_{i,j} \mathbf{P}_{i,j} \log \frac{\mathbf{P}_{i,j}}{\hat{\mathbf{P}}_{i,j}}. \tag{2}$$

By minimizing the above loss function, the agent is encouraged to learn a better map representation for localizing instruction objects.

Instruction-relevant object localization module. The localization module contains two main components, namely a state encoder and a state-instruction attention $Att(\cdot)$. The state encoder is a gated recurrent unit (GRU) [17] and attention module is a scaled dot-product attention [50]. Specifically, we first track visual history, which consists of RGB image R, depth image D, and multi-granularity map $\mathbf{M}$, using the state encoder to encode current episode state $\mathbf{s}_t$. Then, the state $\mathbf{s}_t$ attends to instruction features using the state-instruction attention to generate attended instruction feature $\bar{\mathbf{i}}$, where the instruction features are encoded by a bi-directional LSTM [47] from instruction I. With the attended instruction feature $\bar{\mathbf{i}} \in \mathbb{R}^c$ and a multi-granularity map representation $\mathbf{M} \in \mathbb{R}^{m \times m \times c}$, we predict a localization grid map $\hat{\mathbf{P}} \in \mathbb{R}^{m \times m}$ by calculating the cosine similarity between $\bar{\mathbf{i}}$ and each map region feature in multi-granularity map:

$$\hat{\mathbf{P}} = \mathrm{Softmax}((\mathbf{W}_q \bar{\mathbf{i}})(\mathbf{W}_k \mathbf{M})^T), \tag{3}$$

$$\text{where } \bar{\mathbf{i}} = \mathrm{Att}(\mathbf{s}_t, \mathrm{BiLSTM}(I)), \quad \mathbf{s}_t, \mathbf{h}_t = \mathrm{GRU}([\mathbf{M}, f_R(R), f_D(D)], \mathbf{h}_{t-1}),$$

$\mathbf{W}_q$, $\mathbf{W}_k$ are learnable parameter matrices, $\mathbf{h}_t$ is hidden state of GRU at time step t, $[\cdot, \cdot]$ indicates concatenation operation, $f_R(\cdot)$ and $f_D(\cdot)$ are off-the-shelf RGB and depth encoders, respectively.

3.4 Waypoint navigator and overall learning objective

With the learned multi-granularity map, we feed it together with the instruction to a waypoint navigator for predicting navigation action. Specifically, an attended map features $\bar{\mathbf{m}}$ is produced by weighted averaging multi-granularity map $\mathbf{M}$ using predicted localization results $\hat{\mathbf{P}}$ in the auxiliary task described in Section 3.3. The localization results help agents highlight relevant regions that possibly contain instruction-relevant objects. Then, following CMA [36], another state encoder (GRU) is exploited to predict a state s'_t from the concatenation of attended map features $\bar{\mathbf{m}}$, attended instruction features $\bar{\mathbf{i}}$ and the first state feature $\mathbf{s}_t$, which can be formulated as follows:

$$\mathbf{s}'_t, \mathbf{h}'_t = \mathrm{GRU}([\bar{\mathbf{m}}, \bar{\mathbf{i}}, \mathbf{s}_t], \mathbf{h}'_{t-1}), \tag{4}$$

where $\mathbf{h}'$ is a hidden state of GRU. We feed the state s'_t to a waypoint predictor to predict a waypoint $\hat{\mathbf{w}} = (\Delta x, \Delta y)$ indicating the next navigation goal. We also feed the state to a progress predictor to predict a progress value $\hat{p}$ indicating the completeness of navigation process. Both of these predictors are a fully-connected layer. To get the ground-truth waypoint, we first plan the shortest path from the current agent position to the nearest waypoint on the instruction-relevant path following LAW [44]. Then, we draw a circle with 3 meters radius centered at the agent position. The intersection point between the path and circle is considered as a ground-truth waypoint $\mathbf{w}$.

The ground-truth progress is defined as the normalized distance from the agent current position to the goal following existing work [36]. The waypoint loss l_w and progress loss l_p are defined as follows:

$$l_w = ||\hat{\mathbf{w}} - \mathbf{w}||^2, \quad l_p = ||\hat{p} - p||^2. \tag{5}$$

With a predicted waypoint, we feed it to an off-the-shelf deep reinforcement learning model DD-PPO [52] to determine a low-level action from action space $\mathcal{A}$. If the predicted progress is higher than a threshold λ_p, the agent executes STOP action. We update the waypoint every 3 time steps.

The overall learning objective for our method is as follows:

$$L = l_s + \alpha l_o + \beta l_p + \gamma l_w, \tag{6}$$

where α, β and γ are hyper-parameters.

4 Experiments

4.1 Experimental setups

Dataset and evaluation metrics. We conduct our experiments on VLN-CE dataset, which contains 16,844 path-instruction pairs from 90 scenes in Matterport3D. It also contains about 150k augmented data generated by EnvDrop [48]. The dataset is split into the train, seen validation, unseen validation, and test set. We follow the existing works [22, 36] to evaluate the navigation performance using success rate (SR), oracle success rate (OS), success weighted by path length (SPL), trajectory length (TL), and navigation error from goal (NE). Note that an episode is considered successful if the agent calls STOP action within 3m of the goal. More details about evaluation metrics are put in Appendix.

Implementation details. We implement our method based on Pytorch framework [42] and Habitat[1] simulator [46]. Following existing work [36, 44], the training process consists of two parts, *i.e.*, teaching-force training on augmented data [48] and fine-tuning models using dagger training. We distribute training over 2 NVIDIA V100 GPUs for 3 days on average. A FORWARD action moves the agent forward by 0.25 meters and a TURN action turns by 15°. We use the same set of hyper-parameters as used in the VLN-CE [36] and show these values in Appendix. The U-Net used for extracting fine-grained features is pre-trained separately on RGB observations from Matterport3D scenes following existing work [22]. The map size m is set to 100. α, β, and γ in Equation (6) are set to 10 such that four reward terms are in the same order of magnitude at initialization. Progress threshold λ_p is set to 0.8.

4.2 Comparisons with state-of-the-art methods

We compare our SS-MGMap with current state-of-the-art methods in Table 1 on both seen and unseen validation set of VLN-CE dataset. Our proposed SS-MGMap outperforms all methods that follow the same VLN-CE setting (*i.e.*no panoramic images) in terms of NE, OS, SR, and SPL, which demonstrates the effectiveness of the learned map representation. Besides, we also get comparable results compared with the methods that use panoramic images. This further verifies the importance of the SS-MGMap for helping agents perceive environments comprehensively even just using a camera sensor with a limited field of view. The detailed analyses are as follows.

We categorize existing methods into two types, *i.e.*, one for the methods that without building a map explicitly and one for that with a map representation. For the first type of methods (*e.g.*, Seq2Seq,

[1]https://aihabitat.org/

Table 1: Comparison between our SS-MGMap and state-of-the-art methods on VLN-CE dataset. Methods marked with * use panoramic images. We highlight the best results among the methods that do not use panoramic images in all tables.

	Val-Seen					Val-Unseen				
	TL ↓	NE ↓	OS ↑	SR ↑	SPL ↑	TL ↓	NE ↓	OS ↑	SR ↑	SPL ↑
AG-CMTP* [12]	-	6.60	56.2	35.9	30.5	-	7.90	39.2	23.1	19.1
R2R-CMTP* [12]	-	7.10	45.4	36.1	31.2	-	7.90	38.0	26.4	22.7
HPN+DN* [35]	8.54	5.48	53.0	46.0	43.0	7.62	6.31	40.0	36.0	34.0
CWP-CMA* [28]	11.47	5.20	61.0	51.0	45.0	10.90	6.20	52.0	41.0	36.0
Seq2Seq [36]	9.37	7.02	46.0	33.0	31.0	9.32	7.77	37.0	25.0	22.0
CMA [36]	**9.26**	7.12	46.0	37.0	35.0	8.64	7.37	40.0	32.0	30.0
LAW [44]	9.34	6.35	49.0	40.0	37.0	8.89	6.83	44.0	35.0	31.0
SASRA [30]	8.89	7.17	-	36.0	34.0	**7.89**	8.32	-	24.0	22.0
CM2 [22]	12.05	6.10	50.7	42.9	34.8	11.54	7.02	41.5	34.3	27.6
SS-MGMap (Ours)	10.12	**5.65**	**51.7**	**46.9**	**43.4**	10.00	**6.28**	**47.6**	**38.9**	**34.3**

Table 2: Results on VLN-CE challenge leaderboard. Methods marked with * use panoramic images.

	Test-Unseen				
Team	TL ↓	NE ↓	OS ↑	SR ↑	SPL ↑
CWP-VLNBERT* [28]	13.31	5.89	51	42	36
CWP-CMA* [28]	11.85	6.30	49	38	33
WaypointTeam* [35]	8.02	6.65	37	32	30
VIRL_Team [36]	**8.85**	7.91	36	28	25
CM2 [22]	13.85	7.74	39	31	24
SS-MGMap (Ours)	12.30	**7.11**	**45**	**35**	**28**

CMA, LAW), our method outperforms them by a large margin. Specifically, compared with LAW, which is the strongest baseline among them, we increase the success rate from 40.0% to 46.9% and from 35.0% to 38.9% on val-seen and val-unseen, respectively. We attribute the improvement to the usage of our multi-granularity map, which represents environments explicitly for reducing the learning difficulty. As for the concurrent methods that are also with map representation (*e.g.*, SASRA and CM2), our SS-MGMap also brings a significant improvement against CM2, increasing success rate by 4.0% and 4.6% on val-seen and val-unseen, respectively. We suspect this is because the semantic map used by these methods provides insufficient information for VLN while our SS-MGMap solves this problem. All these results show the effectiveness of the proposed SS-MGMap, which learns a comprehensive map representation for VLN.

We note that our SS-MGMap even outperforms some baselines (*i.e.*, AG-CMTP, R2R-CMTP and HPN+DN) that use panoramic images on val-unseen, increasing success rate from 23.1%, 26.4% and 36.0% to 38.9%, respectively. This show the potential for replacing expensive panoramic camera with a common camera for VLN. We also note that our method have a relatively longer trajectory length. We argue that this is not a good metric for evaluating VLN performance because an unsatisfactory agent who always stops before it reaches the goal will also get a short trajectory length. A better alternative metric is SPL, which considers both episode length and success rate. In terms of SPL, our method significantly outperforms the existing methods who use the same settings with us (37.0% *v.s* 43.4% on val-seen and 31.0% *v.s* 34.3% on val-unseen).

VLN-CE leaderboard. We compare our SS-MGMap with prior work on the held-out test-unseen set used for VLN-CE leaderboard[2]. In Table 2, our method is leading among those that use the standard observation (no panoramas) following VLN-CE settings [36]. For a fair comparison, we only report the results whose manuscripts are publicly available to ensure that no strong prior knowledge (*e.g.*, exploring the environment in prior) is used.

[2]https://eval.ai/challenge/719/leaderboard/1966

Table 3: Ablation study on different granularity information for map representation.

Map Type	Val-Unseen				
	TL ↓	NE ↓	OS ↑	SR ↑	SPL ↑
No Map	11.66	7.20	36.8	26.5	21.4
Fine-Grained Map	10.11	7.11	41.1	31.6	28.2
Semantic Map	10.89	6.80	42.2	33.3	28.2
Multi-Granularity Map (Ours)	**10.00**	**6.28**	**47.6**	**38.9**	**34.3**

Table 4: Ablation study on self-supervised auxiliary task and types of coarse localization ground-truth.

Auxiliary Task	GT Type	Val-Unseen				
		TL ↓	NE ↓	OS ↑	SR ↑	SPL ↑
✗	-	10.24	6.68	41.7	33.1	29.0
✓	Hard	**9.36**	6.36	43.6	35.2	32.3
✓	Soft	10.00	**6.28**	**47.6**	**38.9**	**34.3**

4.3 Ablation studies

Effectiveness of multi-granularity map. Our SS-MGMap contains both fine-grained details and semantic information about environments. We are interested to evaluate whether these different granularity information helps to represent environments for VLN. To this end, we construct two variants, *i.e.*, one only uses the fine-grained map to represent environments and another one only uses a semantic map which is projected from image segmentation results following the current semantic mapping method [22]. In Table 3, our SS-MGMap significantly outperforms these two variants, increasing success rate from 31.6% and 33.3% to 38.9%, respectively. These results show that both granularity information are important for VLN. Without the fine-grained map, agents are hard to recognize object attributes. Without the semantic map, agents are required to recognize all objects, including some common objects that can be represented in the semantic map, from fine-grained object details, which increases learning difficulty. In comparison, our SS-MGMap leverages complementary multi-granularity information for representing diverse objects. We also try a variant that directly takes raw RGB-D as input to perform VLN in an end-to-end manner. However, this variant drops success rate from 38.9% to 26.5%, indicating the importance of building a map explicitly for VLN.

Effectiveness of self-supervised auxiliary task. For teaching agents to learn a representative map that represents diverse objects accurately, we propose a self-supervised instruction-relevant object localization auxiliary task. To evaluate the effectiveness of this task, we remove the object localization loss in Equation 2, so that there is no auxiliary supervision signal for learning correspondence between maps and instructions. In Table 4, this variant performs worse than our SS-MGMap, success rate decreasing significantly from 38.9% to 33.1%. This shows the importance of the proposed self-supervised auxiliary task, which helps to learn map representation from the natural correspondence between map and instruction objects. The coarse ground truth used for this task is soft, *i.e.*, each map region is with a probability from 0 to 1. In this way, agents are required to figure out the degree of correlation between instructions and all map regions. An alternative is a hard binary ground-truth, *i.e.*, map regions whose region-path distance smaller than a threshold are considered positive and other regions are considered negative. In Table 4, this alternative performs better than the baseline that does not use our proposed auxiliary task but worse than our soft localization ground-truth.

Effect of different layer features for fine-grained map. In this paper, we use a U-Net pre-trained for segmentation task as an RGB encoder to extract image fine-grained details. Existing network interpretability work [55, 31] points out that latent features from lower layers are dominated by color and texture concepts while features from higher layers present more object parts. To evaluate which types of features are more representative for describing environment fine-grained details, we project different layer latent features (*i.e.*, first layer, middle layer, last layer, and classification layer) to build fine-grained maps. In Table 5, using last layer features outperforms other variants in terms of almost all metrics. This is reasonable because the last layer features contain information from both the lower layer and higher layer because of the skip connection mechanism in U-Net, which make it more

Table 5: Ablation study on projecting latent features from different layer to build fine-grained map.

Projected Feature	Val-Unseen				
	TL ↓	NE ↓	OS ↑	SR ↑	SPL ↑
First Layer	9.14	6.29	39.6	32.1	28.6
Classification Layer	**9.05**	6.49	40.2	33.3	29.6
Last Layer	10.00	**6.28**	**47.6**	**38.9**	**34.3**

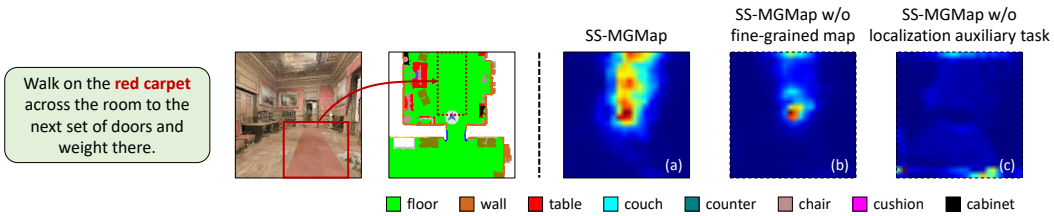


Figure 3: Visualization of instruction-relevant object localization results.

representative for representing environments. The features produced from the classification layer are logits for semantic classification. These features are dominated by semantics information, providing less fine-grained environment details. We also find that an agent using middle layer features cannot be trained to convergence. We suspect it is because the spatial resolution of the middle features (9×9) is too low such that the projected fine-grained map is sparse.

4.4 Visualization results

To evaluate whether the proposed multi-granularity map helps to represent diverse objects, we visualize the localization results $\hat{\mathbf{P}}$ from Equation 3 in Figure 3. Compared with the variant that does not contain a fine-grained map (b) and the variant without the self-supervised auxiliary task (c), our SS-MGMap (a) localizes instruction objects more precisely. Specifically, *red carpet* is incorrectly recognized as *floor* by a segmentation model. With fine-grained map, our SS-MGMap localize it robustly. The variant without fine-grained map is confused to localize it because the semantic map can not represent *red* attribute and *carper* object. The variant without the localization auxiliary task also performs worse because it is hard to establish the correspondence between map representation and diverse instruction-relevant objects. More visualization results are shown in Appendix.

5 Discussion

Limitations and future work. Although our learned multi-granularity map helps for representing environments comprehensively and provides useful information to improve VLN performance, our 2D top-down map is hard to handle the situation when agents go to another floor. Future work may explore to adapt 3D mapping technique [9] to our multi-granularity map. Besides, although our method shows promising performance in photo-realistic simulated data, it has not been thoroughly evaluated in real world. Directing adapting it to real environments may cause accidents such as breaking somethings and hitting pedestrian.

Conclusion. In order to solve the problem that current maps provide insufficient information for VLN task, we propose to gather different granularity information (*i.e.*, fine-grained details and semantic information) for map representation. Moreover, to make the map better represent diverse objects described in instructions, we propose a self-supervised auxiliary task for learning to localize instruction-relevant objects on the map with no need of extra localization annotation. Experimental results show that the proposed method significantly outperforms state-of-the-art methods on VLN-CE benchmark dataset. Qualitative results also show that the proposed multi-granularity map helps to localize instruction objects that are incorrectly classified by current maps.

References

[1] J. Alayrac, A. Recasens, R. Schneider, R. Arandjelovic, J. Ramapuram, J. D. Fauw, L. Smaira, S. Dieleman, and A. Zisserman. Self-supervised multimodal versatile networks. In *NeurIPS*, volume 33, pages 25–37, 2020.

[2] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. D. Reid, S. Gould, and A. van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, pages 3674–3683, 2018.

[3] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. D. Reid, S. Gould, and A. van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, pages 3674–3683, 2018.

[4] R. Arandjelovic and A. Zisserman. Look, listen and learn. In *ICCV*, pages 609–617, 2017.

[5] A. Baevski, W. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. *CoRR*, abs/2202.03555, 2022.

[6] E. Beeching, J. Dibangoye, O. Simonin, and C. Wolf. Learning to plan with uncertain topological maps. In *ECCV*, pages 473–490, 2020.

[7] S. Benaim, A. Ephrat, O. Lang, I. Mosseri, W. T. Freeman, M. Rubinstein, M. Irani, and T. Dekel. Speednet: Learning the speediness in videos. In *CVPR*, pages 9922–9931, 2020.

[8] V. Cartillier, Z. Ren, N. Jain, S. Lee, I. Essa, and D. Batra. Semantic mapnet: Building allocentric semantic maps and representations from egocentric views. In *AAAI*, volume 35, pages 964–972, 2021.

[9] D. S. Chaplot, M. Dalal, S. Gupta, J. Malik, and R. Salakhutdinov. SEAL: self-supervised embodied active learning using exploration and 3d consistency. In *NeurIPS*, pages 13086–13098, 2021.

[10] D. S. Chaplot, D. Gandhi, A. Gupta, and R. R. Salakhutdinov. Object goal navigation using goal-oriented semantic exploration. In *NeurIPS*, pages 4247–4258, 2020.

[11] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta. Neural topological SLAM for visual navigation. In *CVPR*, pages 12872–12881, 2020.

[12] K. Chen, J. K. Chen, J. Chuang, M. Vázquez, and S. Savarese. Topological planning with transformers for vision-and-language navigation. In *CVPR*, pages 11276–11286, 2021.

[13] P. Chen, D. Huang, D. He, X. Long, R. Zeng, S. Wen, M. Tan, and C. Gan. Rspnet: Relative speed perception for unsupervised video representation learning. In *AAAI*, volume 1, page 5, 2021.

[14] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607, 2020.

[15] X. Chen and K. He. Exploring simple siamese representation learning. In *CVPR*, pages 15750–15758, 2021.

[16] Y. Chen, J. Zhang, and Y. Lou. Topological and semantic map generation for mobile robot indoor navigation. In *ICIRA*, pages 337–347, 2021.

[17] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014.

[18] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, 2019.

[19] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, pages 1422–1430, 2015.

[20] D. Fried, R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell. Speaker-follower models for vision-and-language navigation. In *NeurIPS*, pages 3318–3329, 2018.

[21] G. Georgakis, B. Bucher, K. Schmeckpeper, S. Singh, and K. Daniilidis. Learning to map for active semantic goal navigation. In *ICLR*, 2022.

[22] G. Georgakis, K. Schmeckpeper, K. Wanchoo, S. Dan, E. Miltsakaki, D. Roth, and K. Daniilidis. Cross-modal map learning for vision and language navigation. In *CVPR*, 2022.

[23] J. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. Á. Pires, Z. Guo, M. G. Azar, B. Piot, K. Kavukcuoglu, R. Munos, and M. Valko. Bootstrap your own latent - A new approach to self-supervised learning. In *NeurIPS*, pages 21271–21284, 2020.

[24] P.-L. Guhur, M. Tapaswi, S. Chen, I. Laptev, and C. Schmid. Airbert: In-domain pretraining for vision-and-language navigation. In *ICCV*, pages 1634–1643, 2021.

[25] W. Hao, C. Li, X. Li, L. Carin, and J. Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *CVPR*, pages 13134–13143, 2020.

[26] W. Hao, C. Li, X. Li, L. Carin, and J. Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *CVPR*, pages 13134–13143, 2020.

[27] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020.

[28] Y. Hong, Z. Wang, Q. Wu, and S. Gould. Bridging the gap between learning in discrete and continuous environments for vision-and-language navigation. In *CVPR*, 2022.

[29] Y. Hong, Q. Wu, Y. Qi, C. R. Opazo, and S. Gould. VLN BERT: A recurrent vision-and-language BERT for navigation. In *CVPR*, pages 1643–1653, 2021.

[30] M. Z. Irshad, N. C. Mithun, Z. Seymour, H. Chiu, S. Samarasekera, and R. Kumar. SASRA: semantically-aware spatio-temporal reasoning agent for vision-and-language navigation in continuous environments. *CoRR*, abs/2108.11945, 2021.

[31] M. A. Islam, M. Kowal, P. Esser, S. Jia, B. Ommer, K. G. Derpanis, and N. Bruce. Shape or texture: Understanding discriminative features in cnns. In *ICLR*, 2022.

[32] M. G. Jadidi, J. V. Miró, and G. Dissanayake. Mutual information-based exploration on continuous occupancy maps. In *IROS*, pages 6086–6092, 2015.

[33] K. D. Katyal, A. Polevoy, J. L. Moore, C. Knuth, and K. M. Popek. High-speed robot navigation using predicted occupancy maps. In *ICRA*, pages 5476–5482, 2021.

[34] B. Korbar, D. Tran, and L. Torresani. Cooperative learning of audio and video models from self-supervised synchronization. In *NeurIPS*, pages 7774–7785, 2018.

[35] J. Krantz, A. Gokaslan, D. Batra, S. Lee, and O. Maksymets. Waypoint models for instruction-guided navigation in continuous environments. In *ICCV*, pages 15142–15151, 2021.

[36] J. Krantz, E. Wijmans, A. Majumdar, D. Batra, and S. Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *ECCV*, pages 104–120, 2020.

[37] D. Luo, C. Liu, Y. Zhou, D. Yang, C. Ma, Q. Ye, and W. Wang. Video cloze procedure for self-supervised spatio-temporal learning. In *AAAI*, volume 34, pages 11701–11708, 2020.

[38] C. Ma, J. Lu, Z. Wu, G. AlRegib, Z. Kira, R. Socher, and C. Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *ICLR*, 2019.

[39] A. Majumdar, A. Shrivastava, S. Lee, P. Anderson, D. Parikh, and D. Batra. Improving vision-and-language navigation with image-text pairs from the web. In *ECCV*, pages 259–274, 2020.

[40] X. Meng, N. D. Ratliff, Y. Xiang, and D. Fox. Scaling local control to large-scale topological navigation. In *ICRA*, pages 672–678, 2020.

[41] J. O'keefe and L. Nadel. *The Hippocampus as a Cognitive Map*. Oxford university press, 1978.

[42] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8024–8035, 2019.

[43] S. K. Ramakrishnan, Z. Al-Halah, and K. Grauman. Occupancy anticipation for efficient exploration and navigation. In *ECCV*, pages 400–418, 2020.

[44] S. Raychaudhuri, S. Wani, S. Patel, U. Jain, and A. X. Chang. Language-aligned waypoint (LAW) supervision for vision-and-language navigation in continuous environments. In *EMNLP*, pages 4018–4028, 2021.

[45] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241, 2015.

[46] M. Savva, J. Malik, D. Parikh, D. Batra, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, and V. Koltun. Habitat: A platform for embodied AI research. In *ICCV*, pages 9338–9346, 2019.

[47] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681, 1997.

[48] H. Tan, L. Yu, and M. Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL-HLT*, pages 2610–2621, 2019.

[49] E. C. Tolman. Cognitive maps in rats and men. *Psychological review*, 55(4):189, 1948.

[50] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[51] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y. Wang, W. Y. Wang, and L. Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *CVPR*, pages 6629–6638, 2019.

[52] E. Wijmans, A. Kadian, A. Morcos, S. Lee, I. Essa, D. Parikh, M. Savva, and D. Batra. DD-PPO: learning near-perfect pointgoal navigators from 2.5 billion frames. In *ICLR*, 2020.

[53] D. Xu, J. Xiao, Z. Zhao, J. Shao, D. Xie, and Y. Zhuang. Self-supervised spatiotemporal learning via video clip order prediction. In *CVPR*, pages 10334–10343, 2019.

[54] L. Zhang, L. Wei, P. Shen, W. Wei, G. Zhu, and J. Song. Semantic SLAM based on object detection and improved octomap. *IEEE Access*, 6:75545–75559, 2018.

[55] B. Zhou, D. Bau, A. Oliva, and A. Torralba. Interpreting deep visual representations via network dissection. *IEEE transactions on pattern analysis and machine intelligence*, 41(9):2131–2145, 2018.

[56] F. Zhu, Y. Zhu, X. Chang, and X. Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *CVPR*, pages 10009–10019, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Section 5
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Section 5
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] We will publish our code upon acceptance.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4.1.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] Error bars are not reported because it would be too computationally expensive.

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Section 4.1.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]