






LLM as Copilot for Coarse-grained Vision-and-Language Navigation

Yanyuan Qiao¹, Qianyi Liu^{2,3}, Jiajun Liu^{4,5}, Jing Liu^{2,3}, and Qi Wu^{1*}

¹ Australian Institute for Machine Learning, The University of Adelaide

² Institute of Automation, Chinese Academy of Sciences

³ School of Artificial Intelligence, University of Chinese Academy of Sciences

⁴ CSIRO Data61

⁵ The University of Queensland

{yanyuan.qiao,qi.wu01}@adelaide.edu.au, liuqianyi2022@ia.ac.cn,
jiajun.liu@csiro.au, jliu@nlpr.ia.ac.cn

Abstract. Vision-and-Language Navigation (VLN) involves guiding an agent through indoor environments using human-provided textual instructions. Coarse-grained VLN, with short and high-level instructions, has gained popularity as it closely mirrors real-world scenarios. However, a significant challenge is these instructions are often too concise for agents to comprehend and act upon. Previous studies have explored allowing agents to seek assistance during navigation, but typically offer rigid support from pre-existing datasets or simulators. The advent of Large Language Models (LLMs) presents a novel avenue for aiding VLN agents. This paper introduces VLN-Copilot, a framework enabling agents to actively seek assistance when encountering confusion, with the LLM serving as a copilot to facilitate navigation. Our approach includes the introduction of a confusion score, quantifying the level of uncertainty in an agent’s action decisions, while the LLM offers real-time detailed guidance for navigation. Experimental results on two coarse-grained VLN datasets show the efficacy of our method.

Keywords: Vision-and-Language · Navigation · Large Language Models

1 Introduction

The Vision-and-Language Navigation (VLN) tasks have attracted great attention from researchers, which enables an agent to navigate to a specified destination indoors based on the provided textual instructions and visual observations. The tasks of VLN show great potential in real-world applications such as home assistant robots. Most VLN tasks such as R2R [3] and RxR [17] *etc.* provide fine-grained step-by-step instructions, such as “Go out the door in front of you across the room. Once in the hallway turn Left. Walk forward and then turn left into the sitting area. Stop in right in front of the fireplace.” However, though these detailed instructions could improve the agents’ ability of instructions following, they impede the further application of VLN in the real world since human beings would not like to give such long instructions to a robot in daily life. Contrastive to these tasks, the coarse-grained instruction task represented by the Remote

* Corresponding author

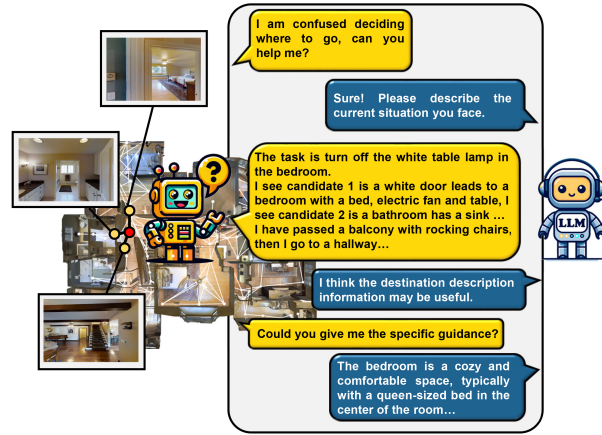


Fig. 1: Our proposed VLN-Copilot. When an agent is confused about making an action decision, it will proactively seek help from the LLM, and then the LLM will assist the agent by analyzing the situations faced by the agent and providing appropriate guidance.

Embodied Referring Expression (REVERIE) task [29] is more expected to enhance the practical utility of VLN because it involves instructions that are closer to those encountered in real-world scenarios, such as “Go clean the mirror in the bathroom.” Giving such coarse-grained instructions to the agent would bring more challenges to the agent’s navigation than fine-grained instructions.

VLN Agents navigating in environments can benefit from seeking external assistance [34]. Early works [27, 37] tried to let agents seek help from oracles or simulators, but the assistance content obtained was fixed and limited. Since this assistance information is based on pre-built ground truth information in datasets or simulators, which are not consistent with real-life scenarios. Inspired by the development of Large Language Models (LLMs) with rich internalized world knowledge, many works have been made to exploit their capabilities to aid navigation tasks. One recent method named MiC [31] makes the first step to use LLM in the coarse-grained VLN task of REVERIE, which queries the LLM to generate more detailed instructions by prompting the LLM with extra planning examples. The promising performance of MiC shows the great potential to use LLM in coarse-grained VLN tasks. However, MiC has some weaknesses in applying LLM for VLN. First, the way the agent seeks help from the LLM is a passive way that depends on the change of the room where the agent is located. This approach may not help the agent when it faces difficulties in making predictions while still in the same room. Second, MiC generates fine-grained planning no matter what challenges the agent faces. This practice lacks a thorough analysis of the difficulties the agent may have and fails to give the agent diverse guidance for different scenarios.

To address these issues, we propose a novel method named VLN-Copilot that makes the LLM a competent copilot to assist the agent in navigation. As shown in Figure 1, before making a movement, the agent calculates a confusion score to assess the level of

uncertainty associated with its current action choice. If this score surpasses a predefined threshold, it triggers the agent to seek guidance. This practice is more proactive than relying on the change of scenes which may be wrong. When the agent decides to seek assistance, it consults the LLM and then the LLM conducts a comprehensive evaluation of the situation, identifies the type of guidance required, and generates the necessary guidance to aid the agent in navigating effectively. The types of guidance information are selected from the in-depth pre-analysis of prevailing challenges the agent may face in the sampled environment using LLM.

We conduct experiments on the REVERIE and CVDN-target dataset and our model surpasses the previous methods. Specifically, our VLN-Copilot obtains 42.32% on the main navigation metric SPL and 26.55% on the main object grounding metric RGSPL on the REVERIE dataset, and 4.47 in GP on the CVDN-target dataset, respectively.

In summary, our contributions are as follows:

- We propose a novel method named VLN-Copilot which utilizes LLM to analyze the difficulties the agent may meet and give diverse guidance for different situations, rather than to generate planning by prompting LLM with extra planning examples.
- We introduce the confusion score to help the agent proactively ask the LLM for help when confused to make predictions rather than passively trigger conversations.
- We utilize LLM with elaborately designed prompts to conduct a detailed challenge analysis of the agent’s different situations, enabling the LLM to provide better guidance on the fly.
- Extensive experiments on two VLN benchmarks show the effectiveness of our proposed method.

2 Related Work

2.1 Vision-and-Language Navigation

In recent years, Vision-and-Language Navigation (VLN) has attracted widespread attention due to its promising applications in home assistant robots and related fields [9, 12, 18, 20, 28, 40]. Many different tasks have emerged in the field of VLN [4, 14, 16, 25, 27, 29, 37]. In terms of the navigation space accessible to the agent, these cover indoor [3, 16, 17], outdoor [4, 35], and aerial scenarios [8, 25].

Early VLN task R2R [3] started from indoor environments requiring the agent to reach the destination according to the given instructions, and there were some subsequent tasks (e.g., RxR [17] and VLN-CE [16]). These tasks all provide detailed natural language instructions, such as “Walk down the stairs at the bottom of the stairs take a left. Walk straight to get to the kitchen inside and wait just inside the door.” These detailed instructions guide the agent through complex step-by-step processes and are more focused on testing the agent’s ability to understand and follow instructions. Some other tasks adopt an object-oriented perspective, such as REVERIE [29] and SOON [44]. These tasks require not only reaching a designated destination but also discovering specific target objects. Different from SOON providing detailed instructions and object recognition prompts, such as “The cabinet is square, closed and wooden. The cabinet is placed below the sink and the water-tap. There are some stools and table nearby.”,

REVERIE’s instructions appear more concise and natural, such as “Bring me the swivel chair from the office.” These coarse-grained instructions, akin to instructions encountered in real-world scenarios, enhance the potential for innovative research within the domain of coarse-grained instruction VLN. However, the task remains challenging due to the limited information these instructions provide.

2.2 VLN Agents Asking for Help

Agents can gain advantages by seeking external assistance to assist navigation, and there have been studies promoting navigation agents to seek help during navigation tasks [27, 34]. Thomason *et al.* [37] introduce the Navigation from Dialog History (NDH) task, which contains dialog history between agents and oracles during navigation. The agent follows guide instructions and asks questions to the oracle when needed. The task provides a valuable resource for investigating the navigation interactions between agent and oracle. Similarly, HANNA [27] builds a simulator where navigation agents can request language-and-vision assistance when they face difficulties during navigation. However, the agent’s signals asking for help are fixed and predefined in the simulator. Additionally, the assistance provided is based on ground truth information derived from the R2R dataset, which may not fully represent real-world navigation scenarios due to its reliance on pre-built datasets. Similarly, Zhu *et al.* proposed SCoA [45], which enables navigation agents to ask questions from a set of pre-constructed candidate question-answer pairs to aid in navigation. Recently, LLM-Planner [36] and MiC [31] have attempted to leverage the knowledge of large language models (LLM) to assist agents in navigation. Agents can query the LLM to generate detailed step-by-step instructions to aid navigation. However, the ways the agents ask for help from both methods are passive, either based on fixed time steps or when the room the agent is in changes. This practice may miss the opportunity to help the agent when it has difficulties in prediction actions. Besides, these two methods concentrate on generating planning by prompting LLM with extra planning data. Different from these methods, we propose a method named VLN-Copilot that allows agents to proactively seek help when they are not sure where to go in this work. Moreover, our VLN-Copilot pays more attention to analyzing the difficulties the agent may meet and can dynamically give appropriate types of guidance in diverse situations for the agent.

3 Method

3.1 Preliminaries

The main goal for the agent in the VLN task is to navigate to a specific destination following the high-level concise instructions. Specifically, the agent is asked to navigate in the simulated environment, which can be described as an undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where \mathcal{V} represents the navigable nodes and \mathcal{E} represents the connectivity edges. Given the natural language instruction L , the agent starts from the initial node V_0 and perceives a panoramic view \mathcal{R}_t of the current node V_t to make the action a_t at each time step t . The panoramic view \mathcal{R}_t has K single-view images r_i and can be represented as

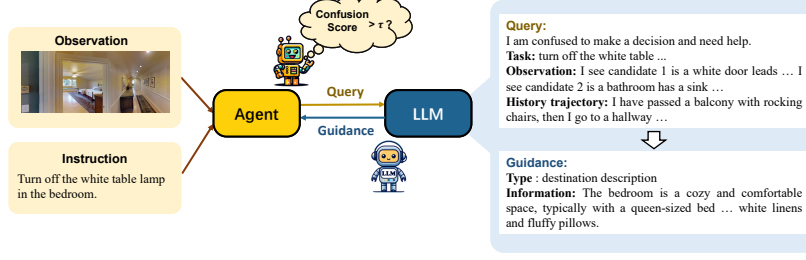


Fig. 2: Overview of the VLN-Copilot. At each time step, the agent predicts the logits over the candidate’s view of actions. After that, the agent will calculate the confusion score to decide whether to ask LLM for help (Sec. 3.3). When the confusion score surpasses threshold τ , the agent queries the LLM for help with the prompt that helps LLM perceive the environment (Sec. 3.4). Then the LLM generates useful guidance information, which will be returned to the agent to assist in decision-making (Sec. 3.6).

$\mathcal{R}_t = \{r_i\}_{i=1}^K$, of which r_i is represented by an image feature vector with the corresponding orientation feature vector. Besides, for the sub-task of object grounding, the object features $\mathcal{O}_t = \{o_i\}_{i=1}^M$ are extracted from the panoramic view \mathcal{R}_t with the annotated object bounding boxes or by object detectors. With these observations of the environment, the agent takes a series of actions $\langle a_0, \dots, a_N \rangle$ to navigate to the target location, where each action a_t selects one navigable node from the candidates. When the target object is grounded or the pre-defined maximum navigation steps have been achieved, the navigation ends.

3.2 Overview

As is shown in Figure 2 and Algorithm 1, given the coarse-grained instruction, the agent first perceives the environment to obtain the current visual observation and starts to predict a probability over candidate actions. Then, a confusion score is computed according to the probability distribution, which determines whether the agent is confused about making predictions in the current situation and whether the agent should ask the LLM for help. If the confusion score exceeds a predefined threshold τ (obtained via grid search) which means the agent is confused about making decisions, the agent initiates a conversation with the LLM to seek their help with an elaborately designed prompt. Specifically, the prompt contains coarse-grained instruction, fine-grained scene descriptions of candidate views, historical descriptions of trajectory scenes, and the requirements for the LLM. Then, the LLM analyzes the agent’s situation and gives advice about which type of guidance information is useful as well as the details about that guidance information. The types of guidance information are refined from the results of challenge pre-analysis by LLM. Finally, the response about the guidance from the LLM is appended after the coarse-grained instruction to guide the agent’s action prediction collaboratively. This paradigm aims to optimize agents’ decision-making processes by

combining instruction, observation, and LLM-based world knowledge via the conversations between the agent and LLM.

Algorithm 1 VLN-Copilot

Notation Summary:

I : high-level instruction
 \mathcal{R}_t : visual observation at timestep t
 S_t : confusion score at timestep t
 W_t^F : fine-grained scene descriptions of all candidate views at timestep t
 W_t^{Tr} : historical trajectory description at timestep t
 T_{guide} : prompt template for LLM to generate guidance information
 LLM: large language model
 τ : pre-defined threshold for confusion score

```

 $t \leftarrow 0$  ▷ Initial timestep
while  $t < \text{max-step}$  and  $\hat{a}_t \neq \text{"stop"}$  do
  if  $S_t \geq \tau$  then
     $P_{\text{guide}} \leftarrow T_{\text{guide}}(I, W_t^F, W_t^{Tr})$ 
     $R_{\text{guide}} \leftarrow \text{LLM}(P_{\text{guide}})$ 
     $I_t^{\text{inter}} \leftarrow \text{CONCAT}(I, R_{\text{guide}})$ 
     $\hat{a}_t \leftarrow \text{Agent}(I_t^{\text{inter}}, \mathcal{R}_t)$ 
  else
     $\hat{a}_t \leftarrow \text{Agent}(I, \mathcal{R}_t)$ 
  end if
   $t \leftarrow t + 1$ 
end while
  
```

3.3 When to Ask LLM for Help

Different from previous methods [31] that set fixed timestep for interactions with LLM or trigger the interactions when the room scenes change, we select a more active practice for the agent by using the confusion score to decide whether to ask for help. Specifically, at each time step t , when the agent is going to predict the logits over the candidate nodes for the next action a_t , we calculate the entropy of the predicted probability distribution over each candidate navigable node v_i as the confusion score S_t :

$$p_t(v_i) = \text{Softmax}(s_t(v_i)), \quad (1)$$

$$S_t = - \sum_i p_t(v_i) \log_2(p_t(v_i) + \epsilon), \quad (2)$$

where $s_t(v_i)$ is the predicted logits assigned by the agent to the i -th navigable node v_i at time step t , and ϵ is a minimal number added to avoid numerical instability.

Then, if the computed confusion score S_t surpasses a predefined threshold τ , which means the agent is confused about making a decision among some candidate nodes, it will send a message to LLM asking for help.

3.4 How LLM Perceives the Environment

LLMs would give inaccurate advice for the agent’s navigation since they are not able to directly acquire visual content from the environment. Both current visual observations from the agent and previously observed scenes are essential for the LLM to analyze the

navigation and supply important information. Thus, enhancing the LLM’s environmental awareness as far as possible is a critical first step to making it a competent copilot. To solve this problem, we translate the dynamic observations and trajectories into multi-grained spatiotemporal descriptions as the eye for the LLM, using the out-of-the-box vision-and-language model of BLIP [19] model. Detailedly, the visually-grounded description mainly consists of three parts: the coarse-grained scene description of the current panoramic view \mathcal{R}_t , the fine-grained scene description of each view r_i of selected candidate navigable nodes at the current time step t , and the trajectory description of previously observed scenes before the current time step t .

Specifically, we obtain both the panoramic view image \mathcal{R}_t of the agent’s observation and the single view image r_i of each candidate navigable node at the time step t . We first feed \mathcal{R}_t to the BLIP model to generate the coarse-grained scene description W_t^C for the agent’s current location. For the fine-grained scene description, the agent is provided with k candidate views for selection. Then, we feed each single-view image r_i of the k selected candidate navigable nodes to BLIP to generate the fine-grained scene description $W_t^{F_i}$:

$$W_t^C = \text{BLIP}(\mathcal{R}_t), \quad (3)$$

$$W_t^{F_i} = \text{BLIP}(r_i), \quad (4)$$

Then we concatenate all previous coarse-grained scene descriptions W^C of each time step before the current time t as the historical trajectory description W_t^{Tr} such as “I passed a bedroom, and then I passed ...”. This process can be formulated as follows:

$$W_t^{\text{Tr}} = \text{CONCAT}(W_1^C, \dots, W_{t-1}^C), \quad (5)$$

Note that we only generate the fine-grained scene descriptions and trajectory descriptions when the agent is confused to make the action prediction and asks the LLM for help, while we generate coarse-grained scene descriptions at each time step. In addition, for better comprehension of LLM, we added the candidate view id i as the prefix to each corresponding native fine-grained scene description $W_t^{F_i}$. For example, we converted the generated scene description of “A room with a tilted ceiling, a bathroom mirror on the wall, and a skylight above the window.” into the candidate view description with the following format: “I see candidate 0 is a room with a tilted ceiling ...”

3.5 Difficulty and Useful Guidance-Type Pre-Analysis

Before asking the agent to query an LLM to generate guidance for assistance, it would be beneficial to explore the difficulties the agent may encounter in making decisions and what guidance information the LLM should provide to be useful for the agent’s navigation. However, manual analysis by humans is costly and impractical. Thus, we turn to LLM for analysis. Specifically, we first collect some navigation samples. When the agent finds it difficult to make decisions during the navigation process, we convert the situation the agent encountered into a text description and save it as a navigation sample. Then, we use an elaborate prompt to query the LLM to analyze the situation encountered by the agent in the current navigation example, such as what is the reason

```

###System: You are a helpful assistant for navigation. There is an agent navigating indoors according to task instructions and current observation. At each step, the agent needs to choose one of the candidate directions to go. I will tell you which direction is correct.

You need to first analyze the current situation, such as what difficulties the agent encountered during navigation that made it difficult to choose a direction and what types of guidance information may be useful.

Task instruction: Turn off the white table lamp in the bedroom
Observation:
I see candidate 1 is a white door leads to a bedroom with a bed, electric fan and table
I see candidate 2 is a bathroom has a sink and the door leads to the shower room
I see candidate 3 is a hallway with white walls, stairs, and a door on the second floor
History trajectory: I have passed a balcony with rocking chairs and a table, then I go to a hallway with a picture of a bedroom.
Correct answer: candidate 1

###Assistant:
Difficulty: The agent encountered difficulties in choosing a direction because there are multiple doors at the same time. This makes it challenging for the agent to determine which direction could find the lamp.

Please list possible type of guidance may be useful.

###Assistant:
Guidance Type: The agent needs to the destination information to make an informed decision.

```

Fig. 3: Prompt template for LLM to pre-analyze the difficulties the agent encounters and what type of guidance information maybe useful. The color of red, blue represent multi-turn responses generated from LLM.

for the difficulty in making a decision, and the helpful types of guidance. Finally, we collect the guidance information generated by LLM and cluster it to obtain the final guidance categories.

Navigation Samples Collection We first let the agent navigate around the environment with confusion scores computed. If the confusion score surpasses the pre-defined threshold at the time step t , which means the agent comes across difficulties in making the action prediction, then we collect the tuple $(I, W_t^F, W_t^{Tr}, A_t)$ of the high-level task instruction I , fine-grained candidate scene descriptions W_t^F , the trajectory description W_t^{Tr} and the ground-truth action A_t as a sample data D_{sample} for analysis. Note that W_t^F includes k fine-grained scene descriptions of the selected candidate views. In total, we randomly collect 5,000 samples.

Prompt Design To make the LLM better analyze the situation that the agent meets, we elaborately designed a prompt template for multi-turn query-and-response. We first set the unified system prompt P_{system} that requires the LLM as a helpful assistant for navigation. Then, we transform each sample data D_{sample} into sample prompt P_{sample} by respectively adding the prefix of “**Task instruction:**”, “**Observation:**”, “**History trajectory:**” and “**Correct answer:**” into I, W_t^F, W_t^{Tr} and A_t . Next, we decompose the task of analyzing the challenge into two sub-tasks, the first of analyzing what difficulties the agent may meet in making decisions and the second of analyzing what types of guidance may be useful for navigation in an iterative way. Correspondingly, we use “**###Assistant: Difficulty:**” and “**###Assistant: Guidance Type:**” as the sub-task prompts P_{diff} and P_{guide} . We concatenate the prompts of P_{system} , P_{sample} and P_{diff} as P_{task1} and feed P_{task1} to LLM, which will generate the response R_{diff} about the difficulty the agent may meet. We further feed P_{task2} to LLM, the concatenation


```

###System: You are a helpful assistant for navigation. There is an agent navigating indoors according to task instructions and current observation. At each step, the agent needs to choose one of the candidate directions to go.

You need to analyze what additional guidance information the agent need to make the choice. There are four types of information: location, prioritization, destination description, previous observations. You can only give one answer from these types.

Task instruction: Turn off the white table lamp in the bedroom
Observation:
I see candidate 1 is a white door leads to a bedroom with a bed, electric fan and table
I see candidate 2 is a bathroom has a sink and the door leads to the shower room
I see candidate 3 is a hallway with white walls, stairs, and a door on the second floor
History trajectory: I have passed a balcony with rocking chairs and a table, then I go to a hallway with a picture of a bedroom.

###Assistant:
Guidance Type: Destination description

Please give description of the destination.

###Assistant:
The bedroom is a cozy and comfortable space, typically with a queen-sized bed in the center of the room. The bed is dressed with crisp, white linens and fluffy pillows.

```

Fig. 4: Prompt template for LLM to generate guidance information.

of P_{task1} , R_{diff} and P_{guide} and we will get the response R_{guide} about what types of guidance may be useful for navigation. The prompt template is depicted in Figure 3.

Guidance-Type Clustering After the above-mentioned operations, we obtain a collection of the guidance response R_{guide} . We first use Sentence-BERT [33] to embed each R_{guide} and get the corresponding embedding E_{guide} . Then we employ the silhouette coefficient [15] to determine the optimal number of clusters for K-Means clustering. The silhouette coefficient is an indicator to evaluate the quality of clustering. It combines the two factors of clustering cohesion and separation, calculates a silhouette coefficient value for each sample, and then calculates a silhouette coefficient value through all samples. By calculating the average silhouette coefficient under different cluster numbers k , the clustering quality under each cluster number can be evaluated. In general, the number of clusters with the highest average silhouette coefficient is considered the optimal number of clusters because it provides the best balance of cohesion and separation. And clustering result shows that the best number of clusters is seven. In addition, after analysis based on the practical situation, we found that some of the guidance content is similar, and some guidance is not executable, such as “Room Layout”, which cannot be obtained from the environment. Thus, we manually merge these duplicate types, remove infeasible types, and finally reserve four applicable categories:

- **Location:** information about the agent’s current position and the destination.
- **Prioritization:** information about the urgent task that needs to be completed based on an assessment of agent progress.
- **Destination Description:** detailed information about the target location, providing more cues to reach its destination.
- **Previous Observations:** information about the agent’s navigation trajectory, lets the agent remember history information.

3.6 LLM Guides the Agent On-the-Fly

After the pre-analysis of the agent’s navigation, we employ the analysis result and a new prompt to ask LLM to aid the agent on the fly when it is confused to make decisions. We first illustrate the modification of the prompt for the on-the-fly guidance. Then, we show how LLM guides the agent with different types of guidance information.

Prompt Design We also designed the prompt template to ask the LLM to analyze the agent’s current situation and decide which guidance type from the selected four categories might be useful in that situation. This prompt template is similar to the pre-analysis prompt template, but with some modifications to guide agent navigation. The full prompt template is shown in Figure 4. First, we modify the system prompt P_{system} by asking the LLM to choose which guidance information is useful from the four pre-selected categories instead of analyzing the difficulties and generating unlimited guidance information. Second, we delete the ground truth action in the sample prompt P_{sample} . At last, the task prompt is revised to “**###Assistant: Guidance Type:**”.

Integrating Guidance with Instruction During the navigation of the agent, when the confusion score S_t surpasses the pre-defined threshold τ at time step t , it will ask the LLM for guidance. With the help of the above-mentioned prompts, the LLM can perceive the environment and trajectory of the agent, analyze the situation, and select the guidance type from the pre-selected four categories. For different types of guidance, the LLM will generate different guidance information I_t^G to assist the navigation. Specifically, the LLM will generate detailed content about the type of “Location”, “Prioritization” and “Destination Description” according to its internalized world knowledge. For these three types of guidance, the generation process is also interactive, which first generates the type of information and then generates the details according to the type of information. Such a two-step process ensures the quality of generated guidance. For the type of “Previous Observations”, we re-use the historical trajectory as the guidance information. Some examples of generated guidance information can be found in Figure 7. Finally, we append the guidance information I_t^G to the high-level instruction I , and then the agent will navigate under the guidance of the integrated instruction I_t^{Inter} .

4 Experiment

4.1 Settings

Datasets We evaluate our method on two VLN tasks: REVERIE and CVDN-target, which are all based upon the MP3D indoor environments. Compared to Room-to-Room (R2R) task, these instructions are high-level and concise, such as “Bring me the swivel chair from the office”. CVDN-target is our proposed VLN setup similar to REVERIE. The instruction contains a target and conversation history. Here we remove the conversation history and only keep the information containing the target as the instruction, such as “To find a book in the room.”

Table 1: Comparison with the state-of-the-art methods on REVERIE.

Methods	Val Unseen						Test Unseen					
	Navigation				Grounding		Navigation				Grounding	
	TL	OSR↑	SR↑	SPL↑	RGS↑	RGSP↑	TL	OSR↑	SR↑	SPL↑	RGS↑	RGSP↑
Human	–	–	–	–	–	–	21.18	86.83	81.51	53.66	77.84	51.44
RecBERT [13]	16.78	35.02	30.67	24.90	18.77	15.27	15.86	32.91	29.61	23.99	16.50	13.51
Airbert [11]	18.71	34.51	27.89	21.88	18.23	14.18	17.91	34.20	30.28	23.61	16.83	13.28
HAMT [5]	14.08	36.84	32.95	30.20	18.92	17.28	13.62	33.41	30.40	26.67	14.88	13.08
VLN-PETL [32]	14.47	37.03	31.81	27.67	18.26	15.96	14.00	36.06	30.83	26.73	15.13	13.03
LANA [39]	16.28	38.54	34.00	29.26	19.03	16.18	16.75	36.41	33.50	26.89	17.53	14.25
HOP+ [30]	14.57	40.04	36.07	31.13	22.49	19.33	15.17	35.81	33.82	28.24	20.20	16.86
DUET [7]	22.11	51.07	46.98	33.73	32.15	23.03	21.30	56.91	52.51	36.06	31.88	22.06
Lily [22]	21.87	53.71	48.11	34.43	32.15	23.43	21.94	60.51	54.32	37.34	32.02	21.94
BEVBert [1]	–	56.40	51.78	36.37	34.71	24.44	–	57.26	52.81	36.41	32.06	22.09
AutoVLN [6]	–	62.14	55.89	40.85	36.58	26.76	–	62.30	55.17	38.88	32.23	22.68
AZHP [10]	22.32	53.65	48.31	36.63	34.00	25.79	–	55.31	51.57	35.85	32.25	22.44
BSG [24]	24.71	58.05	52.12	35.59	35.36	24.24	22.90	62.83	56.45	38.70	33.15	22.34
KERM [20]	21.85	55.21	50.44	35.38	34.51	24.45	17.32	57.58	52.43	39.21	32.39	23.64
ScaleVLN [42]	–	–	56.97	41.84	35.76	26.05	–	–	56.13	39.52	32.53	22.78
GRidMM [41]	23.20	57.48	51.37	36.47	34.57	24.56	19.97	59.55	53.13	36.60	34.87	23.45
MiC [31]	20.64	62.37	56.97	43.60	37.52	28.72	18.11	62.40	55.74	41.97	35.25	26.17
VLN-Copilot	21.89	62.62	57.40	43.63	38.88	29.75	21.72	63.26	57.81	42.32	36.56	26.55

Evaluation Metrics We utilize the standard evaluation metrics to evaluate our method, we report Success Rate (SR), Success weighted by Path Length (SPL) [2], and Goal Progress (GP) as the main navigation metrics, Remote Grounding Success rate (RGS) and RGS weighted by Path Length (RGSP) as the main object grounding metrics.

Implementation Details We implement our proposed method using PyTorch. The model is trained on one single NVIDIA RTX 4090 GPU. For REVERIE, we fine-tuned the pre-trained model from [6] with a batch size of 4 for 100k iterations. For the CVDN-target, we fine-tuned the pre-trained model from [7] with a batch size of 4 for 50k iterations. AdamW [26] optimizer is adopted and the learning rate is set to 1×10^{-5} . The threshold τ is set as 1. For the baseline agent, we use the recent state-of-the-art method of AutoVLN [6]. When collecting data for pre-analysis, we use two agents based on pre-trained AutoVLN [6] and HAMT [5] to avoid agent bias. LLM is integrated into the training process, we use the open-source Vicuna-7B-1.5 model [43], which is fine-tuned from Llama 2 [38].

4.2 Comparison with State-of-the-Art Methods

REVERIE As shown in Table 1, our method achieves competitive performance on the REVERIE benchmark compared to previous methods, such as AutoVLN [6], MiC [31] and ScaleVLN [42]. Particularly, compared with the SoTA method MiC [31], our method outperforms MiC in terms of the object grounding metric RGSP with 1.36% and RGSP with 1.03% on the Val Unseen split. It is worth noting that compared with

Table 2: Comparison on CVDN-target Validation Unseen.

Methods	TL	GP \uparrow
HAMT [†] [5]	37.04	2.74
DUET [†] [7]	78.53	3.70
AutoVLN [†] [6]	61.97	3.98
VLN-Copilot	40.38	4.47

Table 4: Ablation of different threshold τ .

τ	0.8	1	1.2	1.4
SPL \uparrow	42.78	43.63	43.68	42.55
RGSPL \uparrow	28.35	29.75	28.17	28.42

Table 3: Ablation of LLM Analyzer.

Method	Navigation				Grounding	
	TL	OSR \uparrow	SR \uparrow	SPL \uparrow	RGS \uparrow	RGSPL \uparrow
No-Selection (Baseline)	21.96	57.82	52.77	40.15	34.76	26.37
All-Inclusive	20.70	59.76	55.50	41.36	36.47	27.33
Random-Selection	20.15	58.02	53.79	42.16	35.30	27.08
LLM-Selection (Ours)	21.89	62.62	57.40	43.63	38.88	29.75

Table 5: Ablation study of the guidance information.

Method	Navigation				Grounding	
	TL	OSR \uparrow	SR \uparrow	SPL \uparrow	RGS \uparrow	RGSPL \uparrow
Baseline	21.96	57.82	52.77	40.15	34.76	26.37
+ Location	20.97	62.20	56.80	42.97	37.74	28.46
+ Prioritization	20.48	59.47	54.70	41.74	35.86	27.40
+ Destination Description	19.21	59.59	53.99	41.91	34.99	27.11
+ Previous Observations	19.22	57.96	53.73	42.83	35.64	27.12

ScaleVLN [42], which collects 4.9M instruction-trajectory pairs to train the model, our method still achieves better performance (2.8% on SPL and 3.77% on RGSPL).

CVDN-target As shown in Table 2, we compare VLN-Copilot with three main-stream VLN models HAMT [5], DUET [7], and AutoVLN [6] on the CVDN-target dataset. For a fair comparison, both models are trained on the CVDN-target dataset. [†] represents the reproduction results of these methods. It highlights the effectiveness of VLN-Copilot, especially in terms of GP metrics, where it outperforms all other methods. Although TL is slightly higher than HAMT, its significantly improved GP emphasizes its overall advantage in handling unseen validation data.

4.3 Ablation Study

Effect of LLM Analyzer To evaluate the ability of LLM for guidance type selection, we conduct an ablation study, there are four settings, as follows: (1) No-Selection: our baseline, which means the agent does not use any guidance information; (2) All-Inclusive: the agent lets the LLM generate all types of guidance and integrates this information; (3) Random-Selection: the agent randomly selects one of four guidance types and lets the LLM generate corresponding guidance information; (4) LLM-Selection: our method, the LLM outputs guidance type and generates the information. As illustrated in Table 3, we can see that the LLM-Selection method achieved the highest navigation main metric spl with a value of 43.63%, followed by the Random-Selection method with a score of 42.16%. This suggests that blindly combining all types of guidance information is not the best option, probably due to the redundancy of excessively detailed information. Compared to the Baseline, the LLM-Selection method improves the SR by 4.63%, SPL by 3.48%, RGS by 4.12%, and 3.38%.

Different Threshold of Confusion Score As shown in Table 4, we conduct an ablation study on threshold τ in the confusion score. Although when $\tau = 1$, SPL is slightly lower

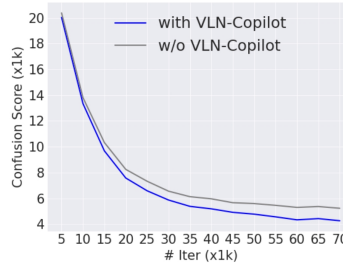


Fig. 5: Confusion Score

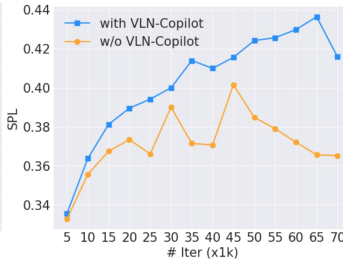


Fig. 6: Learning Curve

than $\tau = 1.2$, while RGSPL reaches a higher value, so considering the two indicators of SPL and RGSPL, we set τ to 1.

Effect of Guidance To validate the effect of different types of guidance information, we conducted an ablation study that only adds one type of information. As shown in Table 5, adding each type of the four categories of guidance information contributes to performance improvement. As shown in Line 1, adding location information contributes mostly, which shows that letting the agent know its own location and target location is of great help. As shown in Line 4, adding previous observations information, the agent’s SPL increased significantly ($2.68\uparrow$), but the SR increased slightly ($0.96\uparrow$). This may be attributed to the agent’s ability to reference historical trajectories, thereby gaining a better understanding of the navigation environment. This, in turn, helps to avoid redundant observations and improve navigation efficiency. Destination Description and Previous Observations reduce TL from 21.96 to 19.21 and 19.22, which shows such two guidance can help the agent reach the destination faster.

Trends of Confusion Score We plot the agent’s cumulative confusion score at each time step every 5k training iterations. As shown in Figure 5, the trend of confusion score is decreasing, especially in the model using VLN-Copilot, the curve is steeper, illustrating that our model could reduce the agent’s confusion degree in navigation.

Learning Curves We compared the learning curves between methods with and without VLN-Copilot during training. As shown in Figure 6, our VLN-Copilot converges faster than the model without Copilot and can achieve better performance (*i.e.* higher SPL value) in fewer iterations.

4.4 Qualitative Results

We present some examples to see the qualitative results of the LLM-generated guidance. As shown in Figure 7, for the example on the left, LLM selects “Destination Description” as the guidance type. Since the guidance information contains a description of the destination and matches candidate direction 2, it is easier to select the answer. For






<p>REVERIE Instruction:</p> <p>Go to the office on level 1 and pick up the phone.</p> <p>Direction Candidates:</p> <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;">  <p>Candidate 1</p> <p>A hallway with a staircase and a door to another</p> </div> <div style="text-align: center;">  <p>Candidate 2</p> <p>A home office with a desk, chair, and pictures on the wall</p> </div> </div> <p>Confusion Score: 1.01</p> <p>Guidance Type by LLM:</p> <p>Destination Description</p> <p>Guidance Information by LLM:</p> <p>The destination is office. It is typically a building or a room with a desk, chair, and other office equipment</p>	<p>REVERIE Instruction:</p> <p>Go to the kitchen and wipe the middle counter top.</p> <p>Direction Candidates:</p> <div style="display: flex; justify-content: space-around;"> <div style="text-align: center;">  <p>Candidate 1</p> <p>You reached the destination.</p> </div> <div style="text-align: center;">  <p>Candidate 2</p> <p>A large refrigerator in a kitchen of a home</p> </div> <div style="text-align: center;">  <p>Candidate 3</p> <p>A kitchen with a stove top oven under a hood</p> </div> </div> <p>Confusion Score: 1.13</p> <p>Guidance Type by LLM:</p> <p>Location</p> <p>Guidance Information by LLM:</p> <p>Your current location is a large kitchen with wooden cabinets and appliances. The destination is the kitchen.</p>
---	---

Fig. 7: Examples of the LLM-generated Guidance.

the example on the right, since Candidate 2 and Candidate 3 are both in the kitchen, and the task is to go to the kitchen, it looks like each candidate is similar, making the choice difficult. While LLM predicts that the guidance type is “location” and provides the agent’s current location and destination, we find that the agent has reached the target location and therefore should stop rather than select a direction to go. More results can be found in supplementary material.

5 Limitations and Future Work

In this work, we utilize Large Language Models (LLMs) to analyze the navigation environment by transforming the dynamic environment into textual descriptions. This practice may be sub-optimal for environmental perception. Our future work will focus on how to effectively apply Multimodal Large Language Models (MLLMs) such as LLaVA [23] and Video-LLaVA [21] to perceive the navigation environment and provide guidance information.

6 Conclusion

In this work, we propose a method named VLN-Copilot that enables the agent to proactively ask for help and lets the LLM act as a co-pilot to assist the agent in completing the coarse-grained VLN task. We first introduce a confusion score calculation mechanism that allows the agent to actively query the LLM when it feels uncertain about the direction of choice. Then, we use LLM to analyze the challenges faced by the agent, and then generate appropriate guidance information for different situations that the agent is in. We conduct extensive experiments on two coarse-grained VLN tasks, including quantitative and qualitative experiments, demonstrating our method’s effectiveness.

Acknowledgements

We thank Mr. Xiangyu Shi for the help in supplying 2 extra NVIDIA RTX 4090 GPUs for experiments.

References

1. An, D., Qi, Y., Li, Y., Huang, Y., Wang, L., Tan, T., Shao, J.: Bevbort: Multimodal map pre-training for language-guided navigation. In: ICCV. pp. 2737–2748 (2023)
2. Anderson, P., Chang, A.X., Chaplot, D.S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., Zamir, A.R.: On evaluation of embodied navigation agents. CoRR [abs/1807.06757](#) (2018)
3. Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I.D., Gould, S., van den Hengel, A.: Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In: CVPR. pp. 3674–3683 (2018)
4. Chen, H., Suhr, A., Misra, D., Snavely, N., Artzi, Y.: TOUCHDOWN: natural language navigation and spatial reasoning in visual street environments. In: CVPR. pp. 12538–12547 (2019)
5. Chen, S., Guhur, P., Schmid, C., Laptev, I.: History aware multimodal transformer for vision-and-language navigation. In: NeurIPS. pp. 5834–5847 (2021)
6. Chen, S., Guhur, P.L., Tapaswi, M., Schmid, C., Laptev, I.: Learning from unlabeled 3d environments for vision-and-language navigation. In: ECCV (2022)
7. Chen, S., Guhur, P.L., Tapaswi, M., Schmid, C., Laptev, I.: Think global, act local: Dual-scale graph transformer for vision-and-language navigation. In: CVPR (2022)
8. Fan, Y., Chen, W., Jiang, T., Zhou, C., Zhang, Y., Wang, X.E.: Aerial vision-and-dialog navigation. In: Findings of the Association for Computational Linguistics: ACL 2023 (2023)
9. Feng, W., Fu, T.J., Lu, Y., Wang, W.Y.: ULN: Towards underspecified vision-and-language navigation. In: EMNLP. pp. 6394–6412 (2022)
10. Gao, C., Peng, X., Yan, M., Wang, H., Yang, L., Ren, H., Li, H., Liu, S.: Adaptive zone-aware hierarchical planner for vision-language navigation. In: CVPR. pp. 14911–14920 (2023)
11. Guhur, P., Tapaswi, M., Chen, S., Laptev, I., Schmid, C.: Airbert: In-domain pretraining for vision-and-language navigation. In: ICCV. pp. 1634–1643 (2021)
12. Hong, Y., Opazo, C.R., Wu, Q., Gould, S.: Sub-instruction aware vision-and-language navigation. In: Webber, B., Cohn, T., He, Y., Liu, Y. (eds.) EMNLP. pp. 3360–3376 (2020)
13. Hong, Y., Wu, Q., Qi, Y., Opazo, C.R., Gould, S.: : A recurrent vision-and-language BERT for navigation. In: CVPR. pp. 1643–1653 (2021)
14. Jain, V., Magalhães, G., Ku, A., Vaswani, A., Ie, E., Baldridge, J.: Stay on the path: Instruction fidelity in vision-and-language navigation. pp. 1862–1872 (2019)
15. Kaufman, L., Rousseeuw, P.J.: Finding groups in data: an introduction to cluster analysis. John Wiley & Sons (2009)
16. Krantz, J., Wijmans, E., Majumdar, A., Batra, D., Lee, S.: Beyond the nav-graph: Vision-and-language navigation in continuous environments. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J. (eds.) ECCV. pp. 104–120 (2020)
17. Ku, A., Anderson, P., Patel, R., Ie, E., Baldridge, J.: Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In: EMNLP. pp. 4392–4412 (2020)
18. Li, J., Tan, H., Bansal, M.: Envedit: Environment editing for vision-and-language navigation. In: CVPR. pp. 15386–15396 (2022)

19. Li, J., Li, D., Xiong, C., Hoi, S.C.H.: Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In: ICML (2022), <https://api.semanticscholar.org/CorpusID:246411402>
20. Li, X., Wang, Z., Yang, J., Wang, Y., Jiang, S.: Kerm: Knowledge enhanced reasoning for vision-and-language navigation. In: CVPR. pp. 2583–2592 (2023)
21. Lin, B., Zhu, B., Ye, Y., Ning, M., Jin, P., Yuan, L.: Video-llava: Learning united visual representation by alignment before projection. arXiv preprint arXiv:2311.10122 (2023)
22. Lin, K., Chen, P., Huang, D., Li, T.H., Tan, M., Gan, C.: Learning vision-and-language navigation from youtube videos. In: ICCV. pp. 8317–8326 (2023)
23. Liu, H., Li, C., Wu, Q., Lee, Y.J.: Visual instruction tuning. arXiv preprint arXiv:2304.08485 (2023)
24. Liu, R., Wang, X., Wang, W., Yang, Y.: Bird’s-eye-view scene graph for vision-language navigation. In: ICCV. pp. 10968–10980 (2023)
25. Liu, S., Zhang, H., Qi, Y., Wang, P., Zhang, Y., Wu, Q.: Aerialvln: Vision-and-language navigation for uavs. In: ICCV. pp. 15384–15394 (October 2023)
26. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR. OpenReview.net (2019)
27. Nguyen, K., III, H.D.: Help, anna! visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) EMNLP. pp. 684–695. Association for Computational Linguistics (2019)
28. Qi, Y., Pan, Z., Hong, Y., Yang, M., van den Hengel, A., Wu, Q.: The road to know-where: An object-and-room informed sequential bert for indoor vision-language navigation. In: ICCV. pp. 1655–1664 (2021)
29. Qi, Y., Wu, Q., Anderson, P., Wang, X., Wang, W.Y., Shen, C., van den Hengel, A.: REVERIE: remote embodied visual referring expression in real indoor environments. In: CVPR. pp. 9979–9988 (2020)
30. Qiao, Y., Qi, Y., Hong, Y., Yu, Z., Wang, P., Wu, Q.: Hop+: History-enhanced and order-aware pre-training for vision-and-language navigation. IEEE TPAMI (2023)
31. Qiao, Y., Qi, Y., Yu, Z., Liu, J., Wu, Q.: March in chat: Interactive prompting for remote embodied referring expression. In: ICCV. pp. 15758–15767 (2023)
32. Qiao, Y., Yu, Z., Wu, Q.: Vln-petl: Parameter-efficient transfer learning for vision-and-language navigation. In: ICCV. pp. 15443–15452 (2023)
33. Reimers, N., Gurevych, I.: Sentence-bert: Sentence embeddings using siamese bert-networks. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) EMNLP-IJCNLP. pp. 3980–3990 (2019)
34. Roman, H.R., Bisk, Y., Thomason, J., Celikyilmaz, A., Gao, J.: Rmm: A recursive mental model for dialog navigation. arXiv preprint arXiv:2005.00728 (2020)
35. Schumann, R., Zhu, W., Feng, W., Fu, T.J., Riezler, S., Wang, W.Y.: Velma: Verbalization embodiment of llm agents for vision and language navigation in street view. In: AAAI (2023)
36. Song, C.H., Wu, J., Washington, C., Sadler, B.M., Chao, W.L., Su, Y.: Llm-planner: Few-shot grounded planning for embodied agents with large language models. arXiv preprint arXiv:2212.04088 (2022)
37. Thomason, J., Murray, M., Cakmak, M., Zettlemoyer, L.: Vision-and-dialog navigation. In: CoRL. pp. 394–406 (2019)
38. Touvron, H., Martin, L., Stone, K.R., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D.M., Blecher, L., Ferrer, C.C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A.S., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I.M., Korenev, A.V., Koura, P.S., Lachaux, M.A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A.,

- Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E.M., Subramanian, R., Tan, X., Tang, B., Taylor, R., Williams, A., Kuan, J.X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., Scialom, T.: Llama 2: Open foundation and fine-tuned chat models. ArXiv **abs/2307.09288** (2023), <https://api.semanticscholar.org/CorpusID:259950998>
39. Wang, X., Wang, W., Shao, J., Yang, Y.: Lana: A language-capable navigator for instruction following and generation. In: CVPR. pp. 19048–19058 (2023)
 40. Wang, X., Huang, Q., Çelikyilmaz, A., Gao, J., Shen, D., Wang, Y., Wang, W.Y., Zhang, L.: Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In: CVPR. pp. 6629–6638 (2019)
 41. Wang, Z., Li, X., Yang, J., Liu, Y., Jiang, S.: Gridmm: Grid memory map for vision-and-language navigation. In: ICCV. pp. 15625–15636 (2023)
 42. Wang, Z., Li, J., Hong, Y., Wang, Y., Wu, Q., Bansal, M., Gould, S., Tan, H., Qiao, Y.: Scaling data generation in vision-and-language navigation. In: ICCV. pp. 12009–12020 (2023)
 43. Zheng, L., Chiang, W.L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., et al.: Judging llm-as-a-judge with mt-bench and chatbot arena. arXiv preprint arXiv:2306.05685 (2023)
 44. Zhu, F., Liang, X., Zhu, Y., Yu, Q., Chang, X., Liang, X.: SOON: scenario oriented object navigation with graph-based exploration. In: CVPR. pp. 12689–12699 (2021)
 45. Zhu*, Y., Weng*, Y., Zhu, F., Liang, X., Ye, Q., Lu, Y., jiao, J.: Self-motivated communication agent for real-world vision-dialog navigation. In: ICCV (2021)