CP3106 Project Report

Vision-Language Navigation with Q&A Interactions

By

Han Yu Xuan

Department of Computer Science

School of Computing

National University of Singapore

2020/04

CP3106 Project Report

Vision-Language Navigation with Q&A Interactions

By

Han Yu Xuan

Department of Computer Science

School of Computing

National University of Singapore

2020/04

Advisor: Dr. Terence Sim
Deliverables:
    Report: 1 Volume

Abstract

Based on an existing study of VNLA (vision-based navigation with language-based assistance), we present a new learning and helping strategy featuring Q&A interactions. The scenario is as follows, (1) a requester gives an agent a command of finding an object in natural language form; (2) the agent receives the command, explores in a photorealistic environment by its egocentric vision, and tries to navigate to the object; (3) the agent may ask questions during this process, and an advisor will provide help by answering the question; (4) the answers, also in natural language form, will be given to the agent to help its navigation. We define a set of questions, train the agent by imitation learning, develop specific metrics, and compare our solutions with the original work. Results show that even though our solution gives a worse performance than VNLA, as our vague and straightforward answers and hints are much less informative, the agent still has learned to ask reasonable questions and leverage the feedback to improve its navigational decisions.

Code available at: https://github.com/xuange666/NUS-CP3106

Video demo available at: https://youtu.be/E7eLIXqEhCQ

Subject Descriptors:

I.2.6 Learning

I.2.8 Problem Solving, Control Methods, and Search

I.2.10 Vision and Scene Understanding

Keywords:

Embodied AI, active learning, navigation, computer vision, natural language processing

Implementation Software and Hardware:

Ubuntu 16.04 LTS, Python 3.7.6, Pytorch 1.0.2, NVIDIA GTX 1080TI with CUDA 10.0

Acknowledgement

I would like to thank Prof. Terence Sim and Mr. Pranavan Theivendiram in guiding me to finish this project.

List of Figures

List of Tables

Table of Contents

Chapter 1

Introduction

1.1 Background

One of the long-standing challenges for AI is to follow human's instructions to solve real-world problems. An ideal home robot, for example, should be able to handle a user's requests like "Hey, can you bring me my laptop on the desk of my bedroom?" Clearly, this is an effortless task for us humans, even if we have never been to that home, as we have common sense about the layout of a bedroom, the appearances of desks and laptops, the indoor structure of a house, etc. For AI, however, it still remains a challenging embodied learning problem with multi-modalities: the AI must learn to understand language, interpret visual content, integrate all sources of information, and finally map them to actions.

1.2 The Problem

A large number of related tasks and solutions have been proposed in recent years (Chen, Jain, Schissler, Gari, Al-Halah, Ithapu, Robinson, & Grauman, 2019; Das, Datta, Gkioxari, Lee, Parikh, & Batra, 2018; Devo, Costante, & Valigi, 2020; Straub, Whelan, Ma, Chen, Wijmans, Green, Engel, Mur-Artal, Ren, Verma, Clarkson, Yan, Budge, Yan, Pan, Yon, Zou, Leon, Carter, Briales, Gillingham, Mueggler, Pesqueira, Savva, Batra, Strasdat, Nardi, Goesele, Lovegrove, & Newcombe, 2019; Jain, Weihs, Kolve, Rastegari, Lazebnik, Farhadi, Schwing, & Kembhavi,

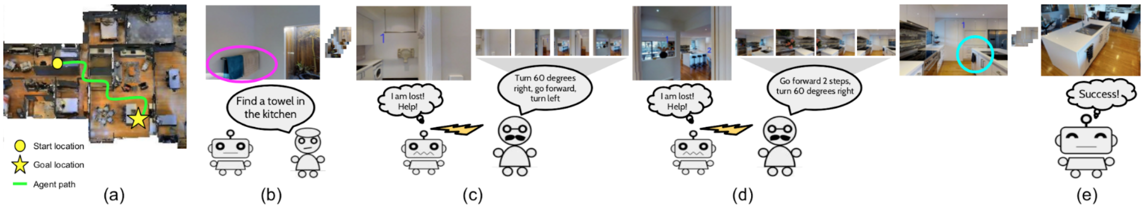
Start location
Goal location
Agent path
(a)
Find a towel in the kitchen
(b)
I am lost! Help!
Turn 60 degrees right, go forward, turn left
1
(c)
I am lost! Help!
Go forward 2 steps, turn 60 degrees right
1
2
(d)
1
Success!
(e)

Figure 1.1: VNLA: Vision-based Navigation with Language-based assistance

2019; Nguyen, Dey, Brockett, & Dolan, 2019), with different constraints and problem settings. Among which, VNLA (Nguyen et al., 2019) proposes to help the AI agent in the goal-driven navigation by providing language support. Specifically, in VNLA, an agent is spawned at a random corner of a house and instructed in the format of "Find *object O* in (one of the) *room R*". During exploration and navigation, the agent can signal the need of help, which is handled by an advisor. The advisor has complete information about the environment, who examines the agent's current position to the goal's position, and provides the agent with the optimal actions for the next several steps. See Figure 1.1 (adopted from (Nguyen et al., 2019)) for illustration.

1.3 Our Solution

Improvements on the original work. One obvious shortage in VNLA is that the interaction between the agent and the advisor is highly restricted and unnatural. The agent, on the one hand, can only say "I am lost! Help!" instead of asking meaningful questions. The advisor, on the other hand, basically tells the agent how to get to the goal with fine-grained action steps, like "turn 60 degrees right, go forward 2 steps, turn left". We consider this kind of "help" as too direct and over-detailed. In our work, we change this kind of interaction into Q&A dialogues. The agent can choose questions to ask from our pre-defined question set, and the advisor only gives simple and short answers accordingly. See Figure 1.2 for illustration. The learning in our settings is more difficult than VNLA, as it is much harder for the agent to utilize the short and straightforward answers compared to the direct descriptions of actions. Besides, the agent not only needs to decide **whether to ask**, given the limited "asking budget", but also decides **what to ask**: which questions are the more reasonable questions that could provide more help

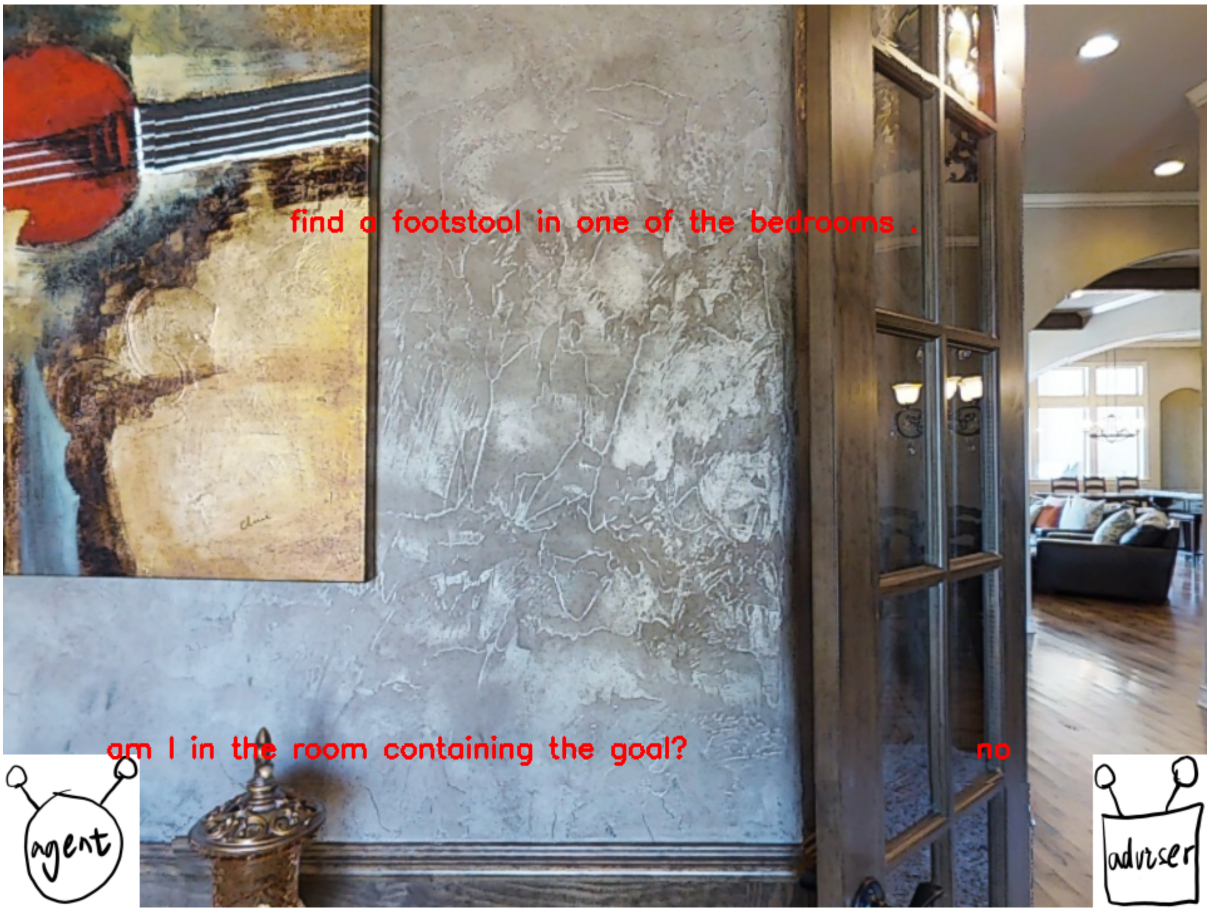
find a footstool in one of the bedrooms .
am I in the room containing the goal?
no
agent
adviser

Figure 1.2: Q&A interaction in vision-language navigation

for navigation under the current situation.

Teachers. As we train the agent by imitation learning, it is also essential to define teachers that make optimal decisions. The policy of the navigation teacher is straightforward: we calculate the shortest path from the agent current position to the closest goal, and derive actions that move along that path. For the asking teacher, we manually design a set of criterions and mappings to decide "whether to ask" and "what to ask". However, these criterions and mappings may not be the "optimum", as will be discussed in 4.1.2, but they provide good enough examples for the agent to follow. See details of the teachers in 3.1.

1.4 Report Organization

For the following sections of the report, we will first do a review over embodied AI, next introduce our Q&A model and learning strategy, then explain the experiment details and finally summarize the results.

Chapter 2

Related Work

2.1 Embodied Cognition

As is mentioned in a neuroscience blog (McNerney, 2011), "Embodied Cognition" is the idea that "the mind is not only connected to the body but that the body influences the mind". In some cases, our mind controls our body on how to behave intelligently, but sometimes the feedback we receive as we interact with the environment (by our body) may also change the way we think (by our mind). An example would be: When frying an egg, our mind tells our body how to break the shell, how often should we turn the egg over, etc. However, after having a taste, our tongue feels it is too salty, which makes our mind learn that we should probably add less salt.

2.2 Internet AI & Embodied AI

In the past years, the focus of AI research is mainly on "Internet AI": passive learning on static datasets and making predictions based on acquired knowledge. It can be viewed as a pure "mind controls body" process. With years of improvements, nowadays deep learning models perform quite well at "Internet AI" tasks, probably because machines are naturally good at memorizing data. In fact, they have already surpassed human capabilities in many areas, such as recognizing objects in images (He, Zhang, Ren, & Sun, 2015) and lip-reading

(Assael, Shillingford, Whiteson, & De Freitas, 2016). Recently, the focus is experiencing a shift to “embodied AI”, active learning and planning within 3D environments. This is where AI agents learn and behave with both processes. However, it turns out that machines are not so good at active explorations and learning from experience, especially in our complicated and unordered real world. One reason could be that machines and programs are always somewhat inflexible, whereas our real world has so many variations.

2.3 Active Learning

One of the main differences between “Internet AI” and “embodied AI” is the learning pattern. In “Internet AI”, the whole training dataset is chunked into batches and fed to the model one by one on every epoch. The model cannot decide “what to learn” and “when to learn”; instead, it examines every sample in the training set in a passive manner. In “embodied AI” however, the model performs active learning. As is described in (Settles, 2009), ”the model is allowed to choose the data from which it learns”. In embodied question answering (Das et al., 2018), for instance, the agent may choose to visit some viewpoints several times while never arriving at other viewpoints.

2.4 Multi-modalities

The information in our world exists in multiple modalities: images, sounds, voices, smells, touches, etc. and we humans have learned to interpret and fuse all these sources for making decisions. In order for AI to carry out tasks in our world, it must also learn to deal with multi-modalities of info. Traditionally, “Internet AI” mainly deals with data in a single modality. Classification, detection and segmentation models only look at pixels; spam filter, sentiment analyzer and translation models only look at text. However, in “embodied AI”, information comes in different modalities, and the model should carry out different subtasks. In vision-language navigation tasks, including goal navigation (Nguyen et al., 2019) and instruction following (Devo et al., 2020), the agent receives a goal or a series of instructions specified by natural language; then it should self-navigate to the goal depending on the top-mounted camera. Note that the agent has no access to the map of the environment. In order to finish such tasks, the agent not only needs to understand the language but also actively explore the environment to gather visual information. Most importantly, it should learn to combine textual and visual information and finally map it into actions that lead the agent to its goal. Embodied question answering (Das et al., 2018), or interactive question answering (Gordon, Kembhavi, Rastegari, Redmon, Fox, & Farhadi, 2018) can be seen as vision-language navigation followed by visual question answering (Antol, Agrawal, Lu, Mitchell, Batra, Lawrence Zitnick, & Parikh, 2015). Furthermore, in some recent work, embodied tasks with even more modalities of information are introduced, such as audio (Chen et al., 2019) and two-agent communications (Jain et al., 2019).

2.5 Common Practices in Embodied AI

Virtual environment. Considering safety issues, experiment cost, and reproducibility, embodied AI experiments are usually carried out in virtual 3D environments. From synthetic environments constructed by game engines such as SUNCG (Song, Yu, Zeng, Chang, Savva, & Funkhouser, 2017) and AI2THOR (Kolve, Mottaghi, Han, VanderBilt, Weihs, Herrasti, Gordon, Zhu, Gupta, & Farhadi, 2017) in some early works, to nowadays photorealistic ones captured by special cameras such as Matterport3D (Chang, Dai, Funkhouser, Halber, Niessner, Savva, Song, Zeng, & Zhang, 2017) and Replica (Straub et al., 2019), an increasing number of 3D environments and simulators are being created. For simplicity, another common practice in experiment is to discretize the environment into blocks (Gordon et al., 2018) or undirected graphs (Nguyen et al., 2019) and also discretize the possible actions of the agent accordingly.

Reinforcement learning. Most of the embodied AI tasks can be formulated into reinforcement learning problems where the agent maintains some policies, acts, transits to another stage and receives rewards from the environment. Particularly, imitation learning (Ross & Bagnell, 2010; Ross, Gordon, & Bagnell, 2011) is often used during training as it is easy to generate expert demonstrations (shortest path to the goal) from the environment. However, such following-shortest-path learning strategies can also result in poor generalization in unseen environments.

Chapter 3

Method

As a large portion of our work is based on VNLA (Nguyen et al., 2019), we describe our method by comparing with it.

3.1 Characters

Both in this work and VNLA, four characters are participating in this whole learning problem: agent, navigation teacher, asking teacher and advisor.

3.1.1 Agent.

The AI agent can be considered as a robot with a top-mounted camera moving around in indoor environments. The agent runs two policies at each time step: navigation policy π_{nav} and π_{ask} asking policy. The navigation policy π_{nav} decides which action to perform next. Specifically, it takes in the attention memory of the end goal G^{main}, the embeddings of the current observation o_t (RGB image), hidden states h_{t-1}, last-step navigation action a^{nav}_{t-1}, asking action a^{ask}_{t-1}, etc., and generates a probability distribution over a set of primitive actions.

$$p^t_{nav} = \pi_{nav}(G^{main}, o_t, h_{t-1}, a^{nav}_{t-1}, a^{ask}_{t-1}), \tag{3.1}$$

$$a_{nav} \in (forward, turn\_left, turn\_right, look\_up, look\_down, stop), \tag{3.2}$$

In VNLA, π_{ask} decides whether to signal the need for help based on the current situation. It also takes in the end goal, observation and etc., and outputs the probabilities of *ask* and *dont_ask*.

In our work, however, as the agent not only needs to decide whether to ask, but also what to ask, π_{ask} outputs a probability distribution over a set of questions instead. (see the details of the questions in Figure 3.2)

$$p_{ask}^{t} = \pi_{ask}(G^{main}, b^{t}, o^{t}, h^{t-1}, a_{nav}^{t-1}, a_{ask}^{t-1}), \tag{3.3}$$

$$a_{ask} \in (arrive, direction, distance, room, dont\_ask), \tag{3.4}$$

3.1.2 Navigation teacher

Both the navigation teacher and the asking teacher have full access to the environment information and are available during training. The navigation teacher can be seen as the optimal navigation policy π_{nav}^{*}, which provides optimal navigation actions for the agent. At each time step, it first finds the closest goal to the agent's current position. (Note that there can be more than one goal in a task, and the agent's arrival at any of the goals will be considered as a success. For example, for command "find a laptop in one of the bedrooms", there may be multiple laptops in different bedrooms). Then it calculates the shortest path to the nearest goal using Dijkstra Algorithm (Dijkstra & others, 1959) (the environment is discretized into an undirected graph) and returns the optimal actions that move along the shortest path.

3.1.3 Asking teacher

The asking teacher can be seen as the optimal asking policy π_{ask}^{*}, which decides whether the agent should request for help based on its internal states and the environment. In VNLA, the asking teacher is heuristic-driven, which decides that the agent should ask for help when any of the manually designed criterions are met. We modify some of the criterions to make them compatible with our Q&A settings. They are denoted as C and listed as follows:

- Deviate: the distance between the current position of the agent and its nearest goal is more than δ meters.
- Uncertain: the agent is "confused", defined as the KL Divergence between the agent's tentative navigation distribution p_{nav} and the uniform distribution is smaller than the

threshold ϵ.

- Unmoved: The agent has remained at the same viewpoint for the last μ steps.
- Same Room: For the last μ' steps, the agent has remained in the same room and has not asked “Same Room” questions.
- Arrive: The agent is at a goal viewpoint or the highest probability action of the tentative navigation distribution is *stop*.

Again, as in our work, the asking teacher should decide not only **whether the agent should ask**, but also **what the agent should ask**, we manually design a questions set Q and a mapping $M1$ to systematically determine which question to ask under which situation.(see Figure 3.2).

$$Q = M1(C) \tag{3.5}$$

It is important to notice that the order of the criterions appeared in Figure 3.2 matters: the asking teacher π^*_{ask} first examines the criterion at the top of this figure (“arrive”). If satisfied, π^*_{ask} would output the corresponding question and ignore all the following criterions. Otherwise, π^*_{ask} examines the next (“deviate”) and continues.

3.1.4 Advisor

The advisor attends to the agent’s requests or answers the agent’s questions. Ideally, the advisor should be a human, who has no map of the environment and answers solely based on common sense. However, for the experiment, we assume the advisor to be another program which has complete information about the environment. In VNLA, when the agent asks for help, the advisor would go and fetch optimal navigation actions for the next k steps from the navigation teacher π^*_{nav}, formulate these k actions into a fine-grained instruction in natural language, and prepend the instruction to the end goal (see Figure 3.1). At the next step the agent takes in the modified version of the end goal and thus gets the instruction. In our work, the advisor answers the agent’s questions by examining the agent’s current position and the environment. On receiving answers, the agent interprets the feedback by another mapping $M2$.

$$I = M2(A) \tag{3.6}$$

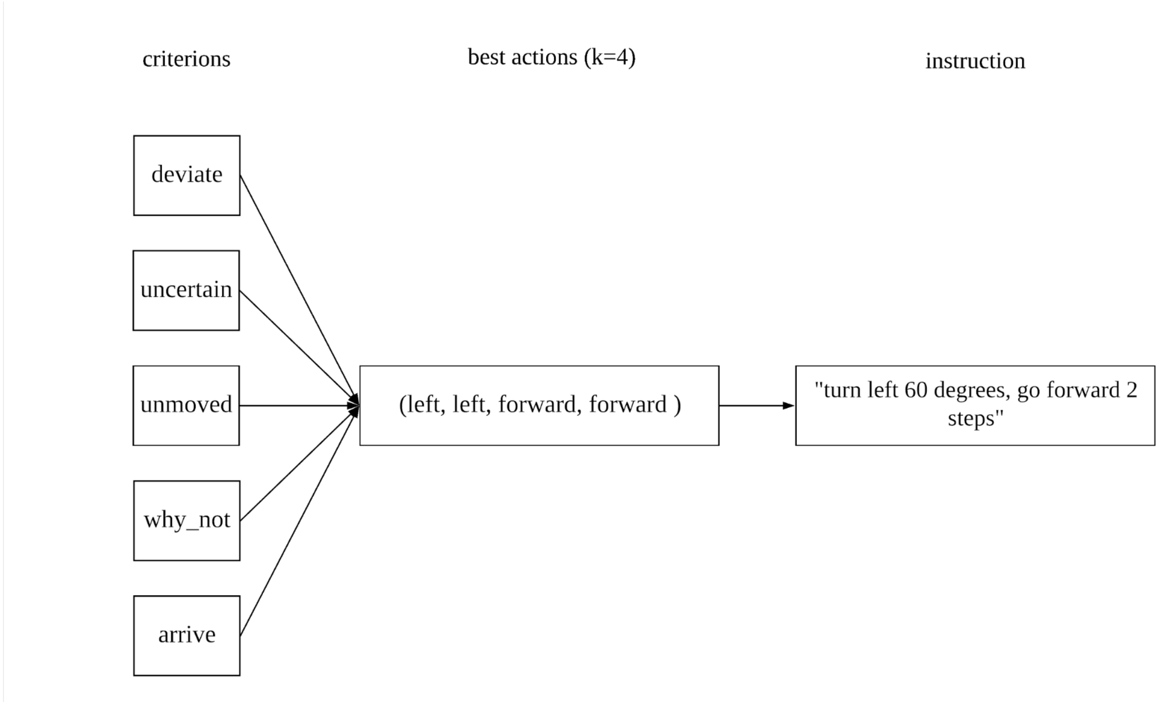

Figure 3.1: Mapping from criterions to instructions in VNLA

$M2$ maps answers A to verbal hints I, which are also prepended to the end goal or replace the end goal (see Figure 3.2).

3.2 Model

Most of the neural architecture in our work is the same as VNLA. As illustrated in Figure 3.3, the whole structure is an encoder-decoder model with a multiplicative attention mechanism (Luong, Pham, & Manning, 2015) and coverage modelling (Tu, Lu, Liu, Liu, & Li, 2016). At step t,

1. The attention memory of the end goal G^{main}, image feature of the current observation o_t (by feeding the current viewpoint of RGB image into a ResNet-152 (He, Zhang, Ren, & Sun, 2016) pretrained on ImageNet (Russakovsky, Deng, Su, Krause, Satheesh, Ma, Huang, Karpathy, Khosla, Bernstein, & others, 2015)), last step hidden states h_{t-1}, last step navigation action a^{nav}_{t-1} and asking action a^{ask}_{t-1} are concatenated and fed into the

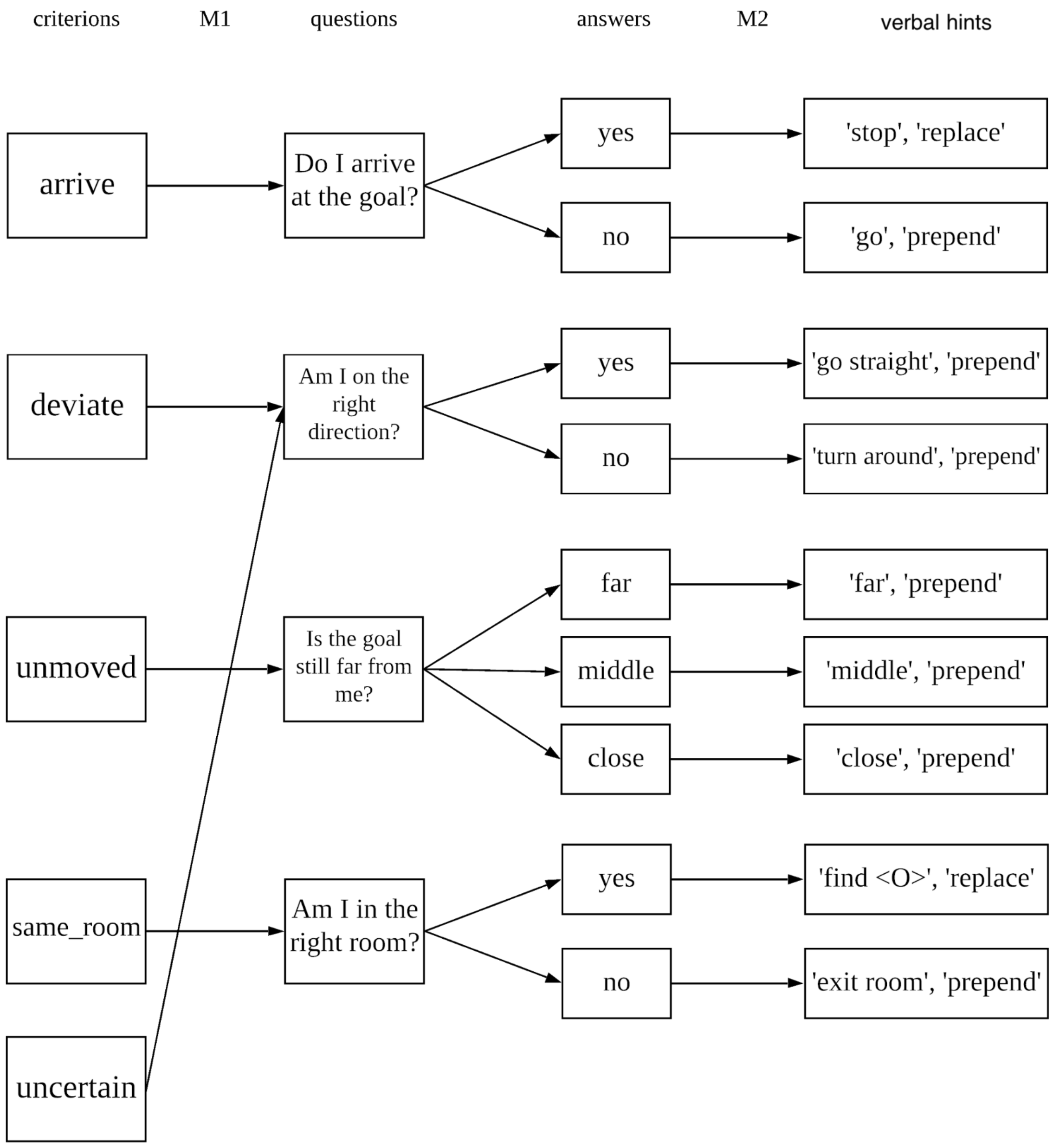

Figure 3.2: Mapping from criterions to verbal hints in our work

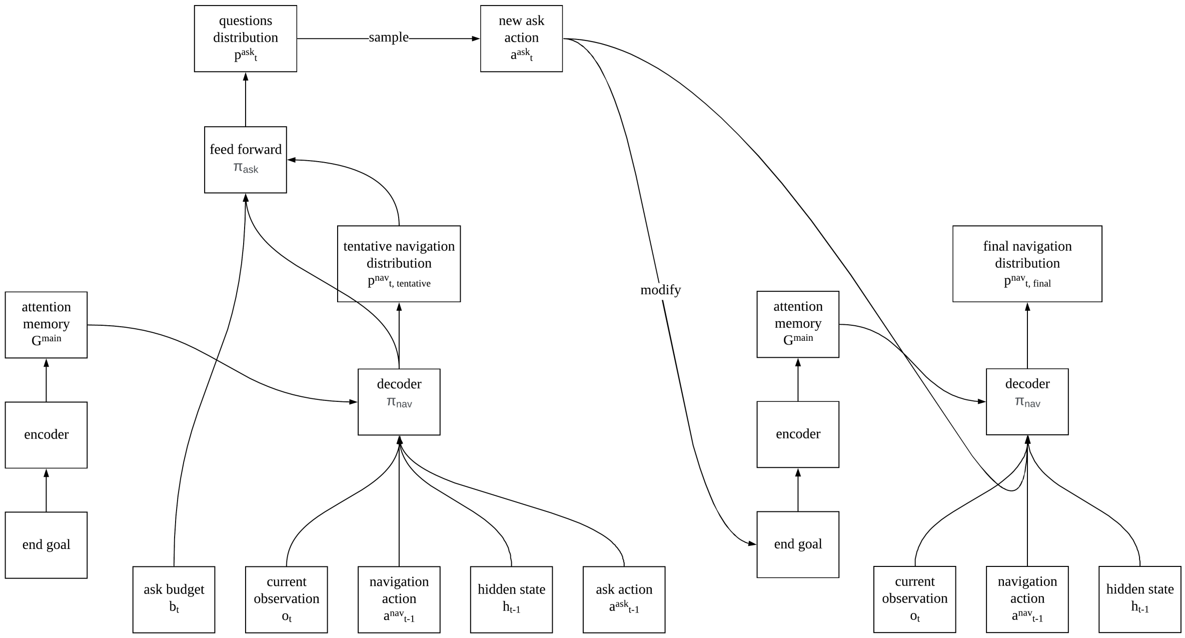
questions distribution p^{ask}_t
sample
new ask action a^{ask}_t
feed forward π_{ask}
tentative navigation distribution $p^{nav}_{t, tentative}$
modify
final navigation distribution $p^{nav}_{t, final}$
attention memory G^{main}
decoder π_{nav}
encoder
end goal
ask budget b_t
current observation o_t
navigation action a^{nav}_{t-1}
hidden state h_{t-1}
ask action a^{ask}_{t-1}

Figure 3.3: Neural architectures for navigation and asking policies

decoder to predict a tentative navigation distribution $p^{nav}_{t,tentative}$.

$$p^{nav}_{t,tentative} = \pi_{nav}(G^{main}, o_t, h_{t-1}, a^{nav}_{t-1}, a^{ask}_{t-1}) \tag{3.7}$$

2. The same input to the decoder, plus the ask budget b_t and the output from step 1 are concatenated and fed to linear layers which generate the question distribution p^{ask}_t.

$$p^{ask}_t = \pi_{ask}(G^{main}, b_t, o_t, h_{t-1}, a^{nav}_{t-1}, a^{ask}_{t-1}, p^{nav}_{t,tentative}) \tag{3.8}$$

3. A new asking action a^{ask}_t is sampled from p^{ask}_t, and the end goal is modified by the advisor accordingly (if a^{ask}_t is not *dont_ask*).

4. The modified goal is again fed to the encoder to get the new encoding, which is concatenated with the observation, hidden states, last-step navigation action and the current asking action and fed to the decoder to get the final navigation distribution $p^{nav}_{t,final}$.

$$p^{nav}_{t,final} = \pi_{nav}(G^{main}, o_t, h_{t-1}, a^{nav}_{t-1}, a^{ask}_t) \tag{3.9}$$

See the detailed description of this process in (Nguyen et al., 2019).

3.3 Training and Inference

We train the agent using imitation learning / supervised learning: the agent learns by mimicking the teachers' behaviors. Specifically, the optimal navigation action a^*_{nav} and asking action a^*_{ask} provided by the teachers π^*_{nav} and π^*_{ask} are used for updating the agent's policies during training. The loss functions are calculated as:

$$L_t^{nav} = cross\_entropy(p_{t,final}^{nav}, a^*_{nav,t}) \tag{3.10}$$

$$L_t^{ask} = cross\_entropy(p_t^{ask}, a^*_{ask,t}) \tag{3.11}$$

Besides, in VNLA, the agent is also trained in a teacher-forcing manner: the agent's decisions are overwritten by optimal actions. For asking actions, $a^*_{ask,t}$ overwrites the sampled result from p_t^{ask} at every time step; for navigation actions, $a^*_{nav,t}$ overwrites the sampled result from $p_{t,final}^{nav}$ if the current time step t is within the next k step of the last ask. This teacher-forcing scheme ensures the coherence between the instruction given to the agent and the agent's actual actions. In other words, the agent's interpretation loss of the instruction is minimized. During inference, however, the asking teacher is no longer available, and the navigation teacher only indirectly influences the agent. See Figure 3.4 and 3.5.

In our work, as the verbal hints are much coarser and less informative, we decide not to replace the agent's decisions by the optimums for both navigation and asking, for the following two reasons:

- It is hard to define corresponding actions for some of the hints like "far", "middle" and "close".

- The agent is given more freedom to explore and figure out its own interpretation of the hints.

Experiments show that the removal of this overwriting design leads to better performance in our Q&A setting.

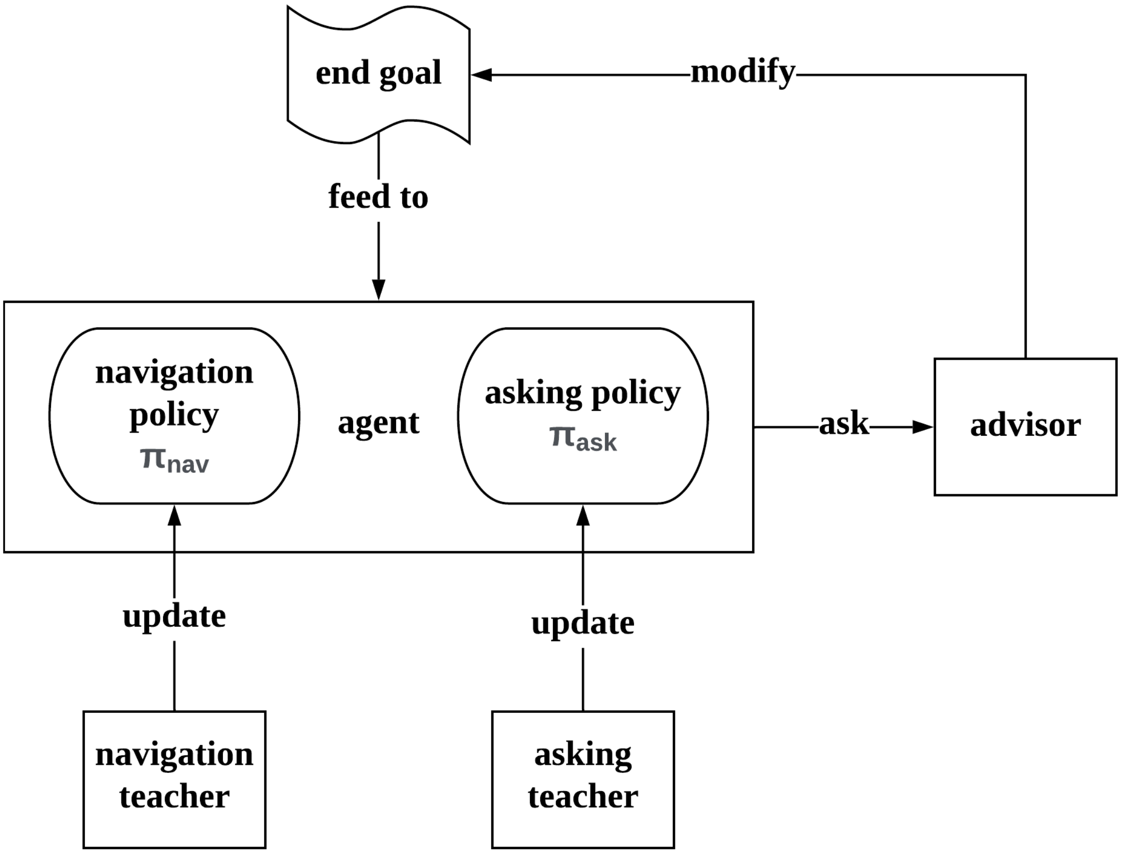

Figure 3.4: Interactions among characters during training

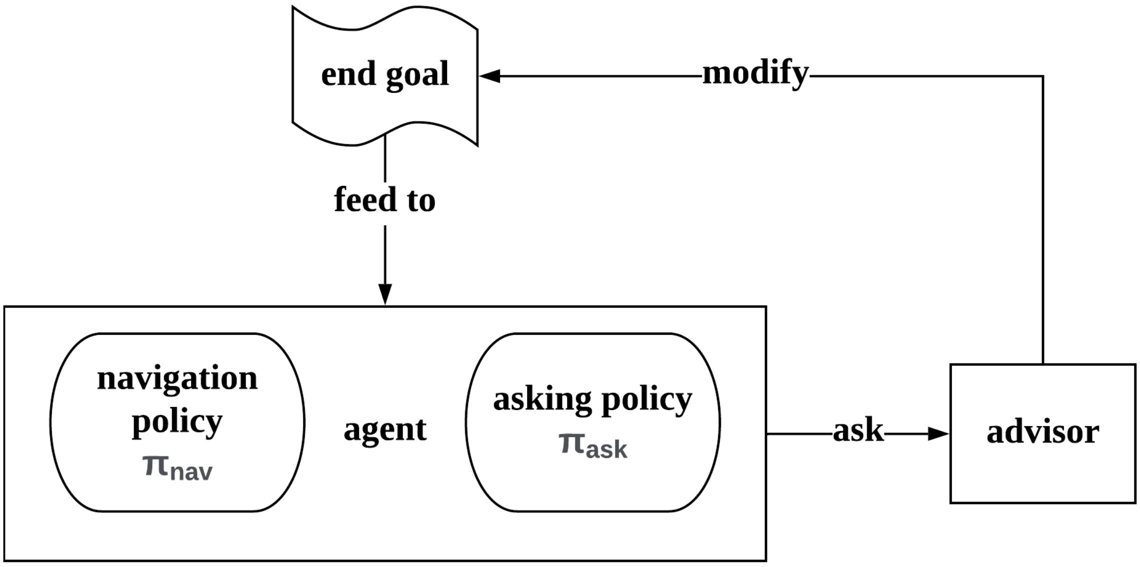

Figure 3.5: Interactions among characters during inference

Chapter 4

Experiment

4.1 Setup

4.1.1 Configurations

All experiments are conducted on a single NVIDIA GTX 1080Ti with CUDA 10.0 in Ubuntu 16.04 LTS. Code written in Python 3.7.6 and Pytorch 1.0.2.

4.1.2 Evaluation Metrics

Evaluation metrics can be divided into 2 categories: navigational and asking. Navigational metrics measures how the agent performs in navigating to goals, which are also the main metrics of this work. Definitions are listed in Table 4.1. Asking metrics measures the quantities and qualities of the questions the agent asks. Definitions are listed in Table 4.2. Notice that in the third asking metric, *bad_questions_per_ep*, we count the number of "bad" questions. A question is considered as "bad" when:

- The agent asks a second or more "am I on the right direction" questions in the same location and with the same heading.
- The agent asks a second or more "is the goal still far from me" questions without much change in the distance to the goal (less than 3 meters).
- The agent asks a second or more "am I the right room" questions in the same room.

Metric Name	**Description**
success_rate	fraction of test set on which the agent stops at a point within d meters from any of the goals
oracle_success_rate	fraction of test set on which any point on the agent's trajectory falls within d meters from any of the goals
room_success_rate	fraction of test set on which the agent stops at any room containing a goal
nav_error	average distance from the point where the agent stops to its closest goal
oracle_nav_error	average of the smallest distances from any point on the agent's trajectory to any goal
length	average distance between where the agent starts and stops
steps	average number of steps the agent takes on each trajectory

Table 4.1: Navigational metrics

Metric Name	**Description**
queries_per_ep	average number of questions asked per episode (per trajectory)
$A/P/R/F$	average accuracy, precision, recall and F1 score (macro) of the question types asked by the agent comparing to the teacher
bad_questions_per_ep	average number of bad questions asked per episode

Table 4.2: Asking metrics

- The agent asks “do I arrive at the goal” question when still far from the goal (more than 5 meters).

We design the *bad_questions_per_ep* metric in addition to $A/P/R/F$, as the latter is not necessarily a proper measurement of the question qualities in our Q&A setting. It only measures how good the agent is at imitating the teacher’s behaviors. Although we manually create some criterions and mappings for the teacher, the resulting policy π^*_{ask} may not be “optimal”. In fact, it is tough to define which question is the “optimal” question at every step. Maybe one question is better than another, i.e., can provide more information in certain situations, but often it is not the only acceptable one. As a result, a good imitator asks high-quality questions, but a lousy imitator does not always ask useless questions. Therefore, we need some manually designed criterions to judge the quality of a question. In experiments, however, we still train the agent by imitating the teacher, and only record but do not punish the agent for asking “bad” questions.

4.1.3 Hyperparameters

Comparing to the hyperparameters in VNLA, we have one more parameter *same_room_threshold*, which corresponds to μ' in the “same room” criterion of the asking teacher. Parameter k, which indicates the number of actions suggested by a subgoal, is rendered invalid thus removed in our Q&A setting. δ, which indicates the deviate threshold, has different context from the original work. To encourage the agent to ask various questions instead of solely relying on a certain type, we lower the uncertainty threshold ϵ to 0.9 and the unmoved threshold μ to 7, so that the probabilities of the asking teacher meeting different criterions are more balanced. Also, as the answers provided contain little information, we increase the asking ratio τ to 0.8, which grants the agent a higher budget for the number of questions it may ask. For all the other unmentioned hyperparameters, they remain the same as VNLA.

Experiment	**Success Rate (%)**	**Room Success Rate (%)**	**Navigation Error (m)**
seen			
vnla	52.26	65.42	3.42
none	28.39	48.97	6.29
ours	42.27	58.76	4.40
unseen			
vnla	34.95	44.85	5.61
none	6.36	14.34	11.30
ours	21.02	29.87	7.44

Table 4.3: Main results compared with VNLA

4.2 Results

We compare our results with VNLA in terms of $success\_rate$, $room\_success\_rate$ and $navigation\_error$. Results are shown separately for the 2 portions of the test set: "seen" and "unseen". In "seen", the environments have been exposed to the agent during training, but end goals are altered. In "unseen", environments are also new to the agent, i.e. they do not appear in the training set. The result in VNLA is referred to as "vnla". Baseline "none", where the agent does not ask any questions, is also added here for comparison. See Table 4.3.

As is expected, the performance of our agent is worse than VNLA, as it is much harder for the agent to learn from our short and vague hints compared to the fine-grained instructions. However, our agent still performs much better than "none", especially in unseen environments. We can infer that the agent has effectively learned to ask questions and interpret hints, which provides much help in fulfilling the navigation task. Results for all other metrics mentioned above are further released in Table 4.4 & 4.5.

Next, we plot out the number of questions asked at different steps in Fig 4.1. Although we have tuned the thresholds to balance question types, the agent still heavily relies on "direction" questions, especially in the first few steps of an episode. As the agent learns by imitating the

Success Rate (%)	**Oracle Success Rate (%)**	**Room Success Rate (%)**	**Navigation Error (m)**	**Oracle Navigation Error (m)**	**Length (m)**	**Step**
seen						
42.27	51.87	58.76	4.40	3.57	11.32	15.31
unseen						
21.02	27.60	29.87	7.44	6.62	8.01	15.79

Table 4.4: Navigational metrics result

Queries Per Episode	**A/P/R/F**	**Bad Questions Per Episode**
seen		
4.0	0.72/0.72/0.46/0.49	0.2
unseen		
7.1	0.67/0.64/0.45/0.45	0.7

Table 4.5: Asking metrics result

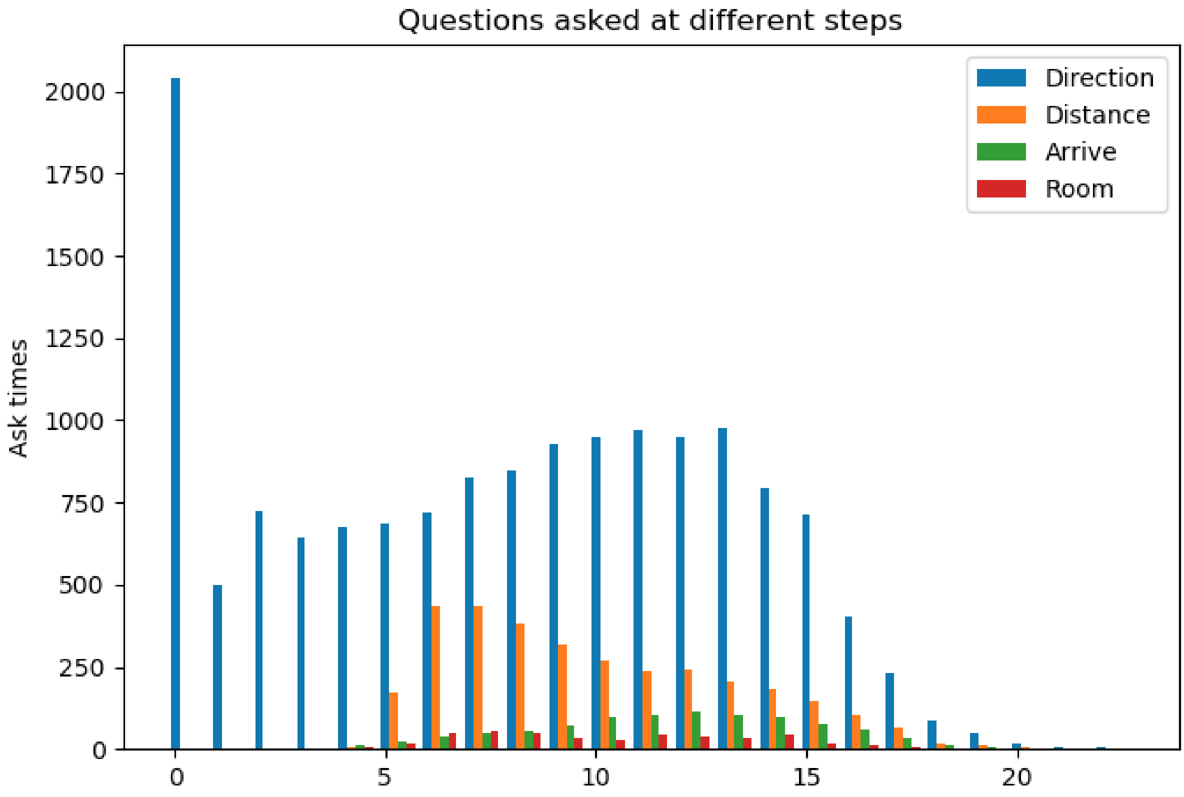

Figure 4.1: Frequencies of questions asked at different steps

teacher's behavior, if we look at the mapping from asking criterions to questions $M1$, it can be inferred that the agent is highly possible to get "confused" at the start of an episode. More "distance" questions appear in the middle of an episode, which means that sometimes the agent may stop moving halfway. Unsurprisingly, "arrive" and "room" questions are seldom seen, as their corresponding criterions are either strict or appear at the lower rank of the priority list.

There is one assumption that the agent has only learned to navigate to the room and happens to stop next to the object. To disprove it, we conduct the same transfer learning experiment in (Nguyen et al., 2019). Here, the training set and training procedure remain the same, however, all the commands in the testing set no longer contain room information, i.e. only object names are provided. Results in Table 4.6 shows that without room information, the agent still achieves comparable performance in seen environments and even higher success rate in the unseen subset.

Experiment	**Success Rate (%)**	**Navigation Error (m)**
seen		
no-room	40.03	4.87
unseen		
no-room	25.93	6.71

Table 4.6: Results on no-room experiment

Chapter 5

Conclusion

5.1 Contributions

In this work, we trained an AI agent via imitating two teachers. The agent has learned to perceive language, navigate in indoor environments, and most importantly, actively ask questions to help its navigational decisions.

5.2 Future Work

We identify three points that can be further improved on.

- Firstly, the way that the asking teacher chooses questions ($M1$) and the way that the agent interprets answers ($M2$) are both fixed mappings. As is discussed in 4.1.2, $M1$ does not necessarily defines the optimal asking policy π^*_{ask}, and $M2$ may not be the only way that the agent utilizes feedback.

- Secondly, the agent does poorly in generalizing to unseen environments, as is shown in the experiment result, there is a huge gap between the agent's performances in seen and unseen subsets. This is an expected result, as we only trained the agent via imitation learning. To remedy the problem, we may define a reward function considering the distance between the agent's position and the goal, and fine-tune the agent using REINFORCE (Mnih, Badia, Mirza, Graves, Lillicrap, Harley, Silver, & Kavukcuoglu, 2016) or A3C (Williams,

1992) algorithms after the imitation learning stage.

- Secondly, questions and answers are all restricted to a small-size pre-defined set. This is because our task, the goal-driven navigation, is rather straightforward and few questions can contribute to better performance. To allow the agent to ask more diversified types of questions, it should be put into more complex situations, especially those where verbal messages can deliver much information and are essential for the success of the task.

References

Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Lawrence Zitnick, C., & Parikh, D. (2015). Vqa: Visual question answering. *Proceedings of the IEEE international conference on computer vision* (pp. 2425–2433), 2015.

Assael, Y. M., Shillingford, B., Whiteson, S., & De Freitas, N. (2016). Lipnet: End-to-end sentence-level lipreading. *arXiv preprint arXiv:1611.01599*, , 2016.

Chang, A., Dai, A., Funkhouser, T., Halber, M., Niessner, M., Savva, M., Song, S., Zeng, A., & Zhang, Y. (2017). Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*, , 2017.

Chen, C., Jain, U., Schissler, C., Gari, S. V. A., Al-Halah, Z., Ithapu, V. K., Robinson, P., & Grauman, K. (2019). Audio-visual embodied navigation. *arXiv preprint arXiv:1912.11474*, , 2019.

Das, A., Datta, S., Gkioxari, G., Lee, S., Parikh, D., & Batra, D. (2018). Embodied question answering. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 2054–2063), 2018.

Devo, A., Costante, G., & Valigi, P. (2020). Deep reinforcement learning for instruction following visual navigation in 3d maze-like environments. *IEEE Robotics and Automation Letters*, *5*(2), 2020, 1175–1182.

Dijkstra, E. W., et al. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, *1*(1), 1959, 269–271.

Gordon, D., Kembhavi, A., Rastegari, M., Redmon, J., Fox, D., & Farhadi, A. (2018). Iqa: Visual question answering in interactive environments. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 4089–4098), 2018.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE international conference on computer vision* (pp. 1026–1034), 2015.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770–778), 2016.

Jain, U., Weihs, L., Kolve, E., Rastegari, M., Lazebnik, S., Farhadi, A., Schwing, A. G., & Kembhavi, A. (2019). Two body problem: Collaborative visual task completion. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6689–6699), 2019.

Kolve, E., Mottaghi, R., Han, W., VanderBilt, E., Weihs, L., Herrasti, A., Gordon, D., Zhu, Y., Gupta, A., & Farhadi, A. (2017). Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, , 2017.

Luong, M.-T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, , 2015.

McNerney, S. (2011). A brief guide to embodied cognition: Why you are not your brain. *Scientific American*, *4*, 2011.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. *International conference on machine learning* (pp. 1928–1937), 2016.

Nguyen, K., Dey, D., Brockett, C., & Dolan, B. (2019). Vision-based navigation with language-based assistance via imitation learning with indirect intervention. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 12527–12537), 2019.

Ross, S., & Bagnell, D. (2010). Efficient reductions for imitation learning. *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 661–668), 2010.

Ross, S., Gordon, G., & Bagnell, D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. *Proceedings of the fourteenth international conference on artificial intelligence and statistics* (pp. 627–635), 2011.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, *115*(3), 2015, 211–252.

Settles, B. (2009). *Active learning literature survey* (Technical report). University of Wisconsin-Madison Department of Computer Sciences.

Song, S., Yu, F., Zeng, A., Chang, A. X., Savva, M., & Funkhouser, T. (2017). Semantic scene completion from a single depth image. *Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition*, , 2017.

Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J. J., Mur-Artal, R., Ren, C., Verma, S., Clarkson, A., Yan, M., Budge, B., Yan, Y., Pan, X., Yon, J., Zou, Y., Leon, K., Carter, N., Briales, J., Gillingham, T., Mueggler, E., Pesqueira, L., Savva, M., Batra, D., Strasdat, H. M., Nardi, R. D., Goesele, M., Lovegrove, S., & Newcombe, R. (2019). The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, , 2019.

Tu, Z., Lu, Z., Liu, Y., Liu, X., & Li, H. (2016). Modeling coverage for neural machine translation. *arXiv preprint arXiv:1601.04811*, , 2016.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, *8*(3-4), 1992, 229–256.