Agent with the Big Picture: Perceiving Surroundings for Interactive Instruction Following

Byeonghwi Kim
GIST
byeonghwikim@gm.gist.ac.kr

Suvaansh Bhambri
GIST
sbhambri@ee.iitr.ac.in

Kunal Pratap Singh
Allen Institute for AI
kunals@allenai.org

Roozbeh Mottaghi
Allen Institute for AI
roozbehm@allenai.org

Jonghyun Choi
GIST
jhc@gist.ac.kr

1. Introduction

We address the *interactive instruction following* task [4, 9, 8] which requires an agent to navigate through an environment, interact with objects, and complete long-horizon tasks, following natural language instructions with egocentric vision. To successfully achieve a goal in the interactive instruction following task, the agent should infer a sequence of actions and object interactions.

When performing actions, a small field of view often limits the agent's understanding of an environment, leading to poor performance. Here, we propose to exploit surrounding views by additional observations from navigable directions to enlarge the field of view of the agent. In addition to the ample observations, while action prediction requires global semantic cues, object localization needs a pixel-level understanding of the environment, making them semantically different tasks. Thus, we design a model factorizing interactive perception and action policy in separate streams in a unified end-to-end framework. The proposed method outperforms the previous challenge winner method [7].

2. Model

The two streams of our model are for action prediction with Action Policy Module (APM) and object localization with Interactive Perception Module (IPM), respectively.

2.1. Interactive Perception Module (IPM)

First, the language encoder in IPM encodes the instructions and generates attended language features. For grounding the visual features to the language features, we use language guided dynamic filters to generate the attended visual features as $h_{t,m} = \text{LSTM}_m([\hat{v}_{t,m}; \hat{x}_{t,m}; a_{t-1}])$, where $[;]$ denote concatenation, $\hat{x}_{t,m}$ and $\hat{v}_{t,m}$ the attended language and visual features, and a_{t-1} the previous action. The class decoder's hidden state $h_{t,m}$ is used to predict the mask m_t.

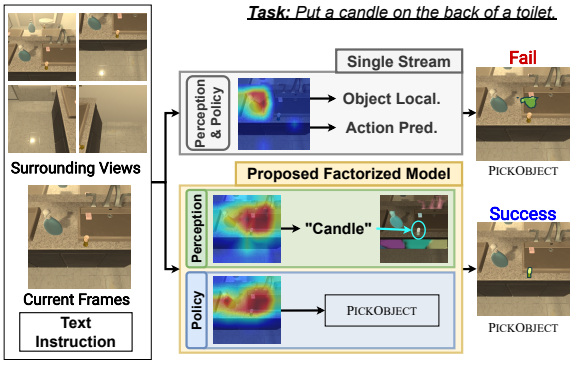

Figure 1: Our model exploits surrounding views and factorizes perception and policy in separate branches. Each heat-map indicates where a stream focuses on in the given visual observation.

Language Guided Dynamic Filters. Visual grounding helps the agent to exploit the relationships between language and visual features. Specifically, the filter generator, f_{DF}, takes the language features, x, and produces N_{DF} dynamic filters. These filters convolve with the visual features, v_t, to output multiple joint embeddings, $\hat{v}_t = \text{DF}(v_t, x)$, as:

$$\begin{aligned} w_i &= f_{DF_i}(x), \;\; i \in [1, N_{DF}], \\ \hat{v}_{i,t} &= v_t * w_i, \\ \hat{v}_t &= [\hat{v}_{1,t}; \ldots; \hat{v}_{N_{DF},t}], \end{aligned} \tag{1}$$

where N_{DF}, $*$ and $[;]$ denote the number of dynamic filters, convolution and concatenation operation respectively.

For richer information, we additionally gather visual features, v_t^1, v_t^2, v_t^3, and v_t^4, from four navigation actions (*i.e.*, left, right, up, and down) including the egocentric visual feature, v_t^0, for each time step. The attended visual feature, $v_{t,m}$, is then comprised of the attended visual feature from each direction as $\hat{v}_{t,m} = [f_{DF}(v_t^0, \hat{x}_{t,m}); \cdots; f_{DF}(v_t^4, \hat{x}_{t,m})]$. Note that our method is not limited to the additional features.

Object-Centric Localization. We bifurcate the task of mask prediction; *target class prediction* and *instance association*. This enables us to leverage the quality of pretrained instance segmentation models with accurate localization.

Target Class Prediction. Our agent first predicts the target object class, c_t, that it intends to interact with at the current time step t, as $c_t = \text{argmax}_k \ \text{FC}_m(h_{t,m}), \quad k \in [1, N_{class}]$, where $\text{FC}_m(\cdot)$ is a fully connected layer and N_{class} denotes the number of the classes of a target object.

Instance Association. When interacting with an object through multiple time steps, it is possible for its appearance to drastically change, causing low confidences of the object's masks. To address such scenarios, we propose a criterion to select the best instance mask. Specifically, the agent predicts the current time step's mask $m_t = m_{\hat{i},c_t}$ with the corresponding center coordinate, $d_t^* = d_{\hat{i},c_t}$, where $\hat{i}$ is:

$$\hat{i} = \begin{cases} \underset{i}{\text{argmax}} \ s_{i,c_t}, & \text{if } c_t \neq c_{t-1} \\ \underset{i}{\text{argmin}} \ ||d_{i,c_t} - d_{t-1}^*||_2. & \text{if } c_t = c_{t-1} \end{cases} \tag{2}$$

Here, d_{i,c_t} and s_{i,c_t} denote the center and the confidence score of a mask instance, m_{i,c_t}, of the predicted class, c_t.

2.2. Action Policy Module (APM)

Same as IPM, we employ the language guided dynamic filters for generating attended visual features. Although we use a similar architecture for IPM, the information captured by dynamic filters is different from that of IPM due to different predictions and hence losses as Equation 3,

$$\begin{aligned} u_a &= [\hat{v}_{t,a}; \hat{x}_{t,a}; a_{t-1}], \quad h_{t,a} = \text{LSTM}_a(u_a) \\ a_t &= \underset{k}{\text{argmax}}(\text{FC}_a([u_a; h_{t,a}]). \quad k \in [1, N_{action}] \end{aligned} \tag{3}$$

where $\hat{v}_{t,a}$, $\hat{x}_{t,a}$ and a_{t-1} denote attended visual features, attended language features, and previous action embedding, respectively. FC_a, takes as input $\hat{v}_{t,a}$, $\hat{x}_{t,a}$, a_{t-1}, and $h_{t,a}$ and predicts the next action, a_t. Note N_{action} denotes the number of actions.

Obstruction Evasion. To address unanticipated situations such as obstacles during inference, we propose an 'obstruction evasion' mechanism in the APM. While navigating in the environment, at every time step, the agent computes the distance between visual features at the current time step, v_t, and the previous time step, v_{t-1} with a tolerance hyper-parameter ϵ as following:

$$d(v_{t-1}, v_t) < \epsilon, \tag{4}$$

where $d(v_{t-1}, v_t) = ||v_{t-1} - v_t||_2^2$. When this equation holds, the agent removes the action that causes the obstruction from the action space so that it can escape.

Split	Model	Seen		Unseen	
		Task	Goal-Cond	Task	Goal-Cond
Val.	Shridhar *et al.* [8]	4.00 (2.10)	10.50 (7.20)	0.20 (0.10)	7.50 (5.10)
	LWIT [7]	33.70 (28.40)	43.10 (38.00)	9.70 (7.30)	23.10 (18.10)
	Ours (ABP)	**42.93** (3.84)	**50.45** (4.76)	**12.55** (1.05)	**25.19** (2.25)
Test	Shridhar *et al.* [8]	3.98 (2.02)	9.42 (6.27)	0.39 (0.08)	7.03 (4.26)
	LWIT [7]	29.16 (24.67)	38.82 (34.85)	8.37 (5.06)	19.13 (14.81)
	LWIT [7]*	30.92 (25.90)	40.53 (36.76)	9.42 (5.60)	20.91 (16.34)
	Ours (ABP)	**44.55** (3.88)	**51.13** (4.92)	**15.43** (1.08)	**24.76** (2.22)

Table 1: **Task and Goal-Cond. Success Rate.** PLW metrics are in parentheses. * denotes the result only on the leaderboard.

Infinite-Loop Evasion. Even though the agent does not encounter the immovable objects, the agent could get stuck in 'infinite-loop' states. Inspired by [3], we adopt an external memory to detect such states. Given the history of visual observations, the agent recognizes the current state as infinite loop if the history contains the two same sub-sequences at least and randomly turns left or tight.

3. Experiments

Dataset and Metrics. To train and evaluate our model on the interactive instruction following task, we use the ALFRED benchmark that runs in AI2-THOR [6]. The scenes in ALFRED are divided into 'train', 'validation' and 'test' sets. We follow the evaluation metrics proposed in [8] (*i.e.*, Success Rate denoted by *Task* and Goal Condition Success Rate denoted by *Goal-Cond*). Additionally, to measure the efficiency of an agent, the above metrics are penalized by the length of the path, denoted by a path-length-weighted (PLW) score for each metric [1].

Implementation Details. The egocentric visual observations are resized to 224×224. For the visual encoder, we use a pre-trained ResNet-18 [5]. For the experimental results, we use the goal statement as input for the IPM and step-by-step instructions for the APM. The model is trained end-to-end using Adam for 30 epochs with an initial learning rate of 10^{-3} with a batch size of 16. We augment visual features by shuffling the channel order of each image and applying image operations following [2].

Results. As shown in the Table 1, the proposed method with surrounding perception outperforms the previous challenge winner [7] on all "Task" and "Goal-Cond" metrics in seen and unseen environments for both validation and test splits by large absolute margins.

4. Conclusion

We explore the problem of interactive instruction following. To address this compositional task, we propose a model that exploits surrounding views and factorizes the task into two streams, interactive perception and action policy, followed by improved components for object localization and obstacle avoidance. Our method provides a framework that can be adopted by future works on ALFRED and beyond.

References

[1] Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R. Zamir. On evaluation of embodied navigation agents. arXiv:1807.06757, 2018. 2

[2] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation policies from data. In *CVPR*, 2019. 2

[3] Heming Du, Xin Yu, and Liang Zheng. Learning object relation graph and tentative policy for visual navigation. In *ECCV*, 2020. 2

[4] Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. Iqa: Visual question answering in interactive environments. In *CVPR*, 2018. 1

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2

[6] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. arXiv:1712.05474, 2017. 2

[7] Van-Quang Nguyen, Masanori Suganuma, and Takayuki Okatani. Look wide and interpret twice: Improving performance on interactive instruction-following tasks. *arXiv:2106.00596*, 2021. 1, 2

[8] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *CVPR*, 2020. 1, 2

[9] Yuke Zhu, Daniel Gordon, Eric Kolve, Dieter Fox, Li Fei-Fei, Abhinav Gupta, Roozbeh Mottaghi, and Ali Farhadi. Visual semantic planning using deep successor representations. In *ICCV*, 2017. 1