AVAST: Attentive Variational State Tracker in a Reinforced Navigator

Je-Wei Jang[1], **Mahdin Rohmatillah**[2], and **Jen-Tzung Chien**[1]

[1]Institute of Electrical and Computer Engineering
[2]EECS International Graduate Program
National Yang Ming Chiao Tung University, Taiwan
{carbon1124.ee08,mahdin.ee08,jtchien}@nycu.edu.tw

Abstract

Recently, emerging approaches have been proposed to deal with robotic navigation problems, especially vision-and-language navigation task which is one of the most realistic indoor navigation challenge tasks. This task can be modelled as a sequential decision-making problem, which is suitable to be solved by deep reinforcement learning. Unfortunately, the observations provided from the simulator in this task are not fully observable states, which exacerbate the difficulty of implementing reinforcement learning. To deal with this challenge, this paper presents a novel method, called as attentive variational state tracker (AVAST), a variational approach to approximate belief state distribution for the construction of a reinforced navigator. The variational approach is introduced to improve generalization to the unseen environment which barely achieved by traditional deterministic state tracker. In order to stabilize the learning procedure, a fine-tuning process using policy optimization is proposed. From the experimental results, the proposed AVAST does improve the generalization relative to previous works in vision-and-language navigation task. A significant performance is achieved without requiring any additional exploration in the unseen environment.[1]

1 Introduction

Reinforcement learning (RL) has become a crucial and successful solution in many sequential decision-making problems, such as video game playing AI (Bellemare et al., 2013) and robotic control (Todorov et al., 2012). In theory, RL algorithms are designed for solving problems under the assumption of Markov decision process (MDP), which means that the observation provided from the environment needs to exactly represent the complete state information of the environment (Chien et al., 2021). However, most of the real-world problems, such as bridge-playing AI, dialogue systems (Rohmatillah and Chien, 2021b; Hsu et al., 2021; Rohmatillah and Chien, 2021a), autonomous driving, and first-person navigation (Kempka et al., 2016), can not be directly modeled as Markov decision processes, because of the incomplete state information. For example, in dialogue task, system does not have an access to the user goal (Jang et al., 2022). In order to improve the generalization, partially observable Markov decision process (POMDP) (Åström, 1965) was designed to model the process in which the agent does not have access to observe complete state information.

In case of vision-and-language navigation (VLN) task, the problem formulation is considered as POMDP problem, as the agent does not receive full information about the state. It only receive the information about the images of surroundings and the texts which describe the navigation task and agent pose information. There is no information which explicitly tells about agent and goal location coordinates. Furthermore, as each observation is unique and complex in the VLN task, the common methods which turn POMDP problem into MDP problem by aggregating the observations and estimating the belief states do not work very well. Aggregation methods usually use either a frame-stacking trick (Mnih et al., 2015) or a recurrent neural network (Hausknecht and Stone, 2015) to aggregate the history observation or the belief state information. These methods mostly work only for either computer vision or natural language processing tasks by considering sufficient information process (Striebel, 1965) assumption as well as Bayes theory (Igl et al., 2018; Lee et al., 2020). Meanwhile, the VLN task requires agent to consider both domains to solve the problem.

Motivated by the aforementioned issues, this work formulates VLN task as a POMDP problem and solves it by using RL algorithms. We propose a

[1] The dataset, simulator and training code are publicly available at: https://github.com/NYCU-MLLab/

Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 424–433
November 20–23, 2022. ©2022 Association for Computational Linguistics

new method named as Attentive VAriational State Tracker (AVAST) to estimate the belief state distribution of the complex observations in the VLN task. AVAST follows sufficient information process assumption to reduce VLN task into an MDP problem. By using variational inference approach, the generalization property of the belief state sampled from AVAST is accordingly held. Based on the experiment result, the proposed method can achieve better performance compared to the baselines due to its generalization property. The organization of this work is arranged as follows. In Sections 2 and 3, the recent approaches to deal with POMDP state tracking and VLN task are discussed, respectively. The proposed method, AVAST, is explained in Section 4. The experimental setup and result are described in Section 5. Finally, Section 6 shows the conclusions.

2 Partially Observable Markov Decision Process State Tracking

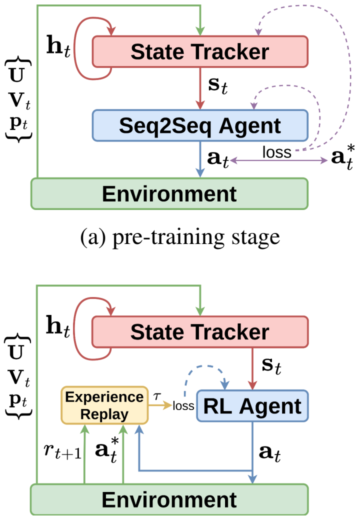
Real-world problems usually cannot directly be modelled as MDP problems, because of the information limitation. Accordingly, the partially observable Markov decision process (POMDP) (Åström, 1965) is fitted to implement an agent decision process in presence of incomplete state information. In general, a POMDP problem can be described by a 6-tuple set $\{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \mathcal{O}\}$. Identical to MDP problem, $\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma$ denote the state, action, transition probability, reward, and discount factor, respectively. The main difference is that the agent can not observe the complete state $\mathbf{s} \in \mathcal{S}$. It only receives an observation $\mathbf{o} \in \Omega$. According to the probability distribution $\mathcal{O}(\mathbf{s})$, the observation $\mathbf{o}$ is generated from the underlying system state as $\mathbf{o} \sim \mathcal{O}(\mathbf{s})$. Generally, estimating a policy distribution from an observation can be arbitrary due to $\pi(\mathbf{a}|\mathbf{o};\phi) \neq \pi(\mathbf{a}|\mathbf{s};\phi)$. Following the sufficient information process (Striebel, 1965), POMDP state distribution can be approximated by using a state tracker to produce the belief state distribution $p(\mathbf{s}|I_t^C)$. I_t^C denotes the complete information state at time t which represents the history information from the beginning to time t. I_t^C is defined as, $I_t^C = \langle \rho(\mathbf{s}_0), \mathbf{o}_1, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}, \mathbf{o}_t \rangle$, where $\rho(\mathbf{s}_0)$ is a distribution over initial stated. Once the well-trained state tracker is obtained, a belief state $\mathbf{s}_t$ can be sampled from the distribution $p(\mathbf{s}|I_t^C)$, and RL agent will consider it as the system state to generate the action $\mathbf{a}_t$.

Traditionally, common sequential learning using recurrent neural network (RNN) was applied to encode the observations history to produce an appropriate belief state as the input to agent (Hausknecht and Stone, 2015). such method was likely to summarize history by remembering features from the past trajectories rather than actually estimating belief states. Furthermore, naively applying RNN would output suboptimal belief states due to the deterministic computation without any distribution constraint. Other approaches (Igl et al., 2018; Lee et al., 2020) estimated the belief states by introducing Bayesian theory. Compared to the purely RNN-based methods, introducing stochastic estimation can improve generalization to complex environments. However, dealing with unseen environment is still a major stumbling block in designing a state tracker. Therefore, different from the previous works, in this paper, an attentive variational state tracker is proposed to improve the state tracking generalization for vision-language navigation.

3 Vision-and-Language Navigation

In general, the reinforcement learning agent which is designed for VLN task (Anderson et al., 2018), will not receive complete state information. Instead, the observation $\mathbf{o} \in \Omega$, generated from the underlying system state according to the probability distribution $\mathbf{o} \sim \mathcal{O}(\mathbf{s})$, will be obtained by the agent in VLN. The observations $\mathbf{o}$ can be separated into three parts which are instructions, visions, and pose information. Instructions are provided in natural language (Chu et al., 2022) to guide the agent about how to reach the target position ρ_{goal} from the initial position ρ_1. At different positions ρ_{t}, agent will receive different panoramic visions and pose information. Given such a process, VLN agent must understand the current situation using the provided instructions, panoramic visions and pose information, and navigate to the target position. Formally, an agent will receive one instruction $\mathbf{U} \in \Omega^u$ at the beginning, and at the same time receive an initial panoramic vision $\mathbf{V}_1 \in \Omega^v$ and an initial pose information $\mathbf{p}_1 \in \Omega^p$, generated from the initial position ρ_1. Then, it will receive a current panoramic vision $\mathbf{V}_t \in \Omega^v$, current pose information $\mathbf{p}_t \in \Omega^p$, and reward $r_t \in \mathcal{R}$, generated from the current position ρ_t at each time step t after acting an action $\mathbf{a}_{t-1}$.

Due to the difficulty of VLN task, the most intuitive way to deal with this task is to apply imita-



(a) pre-training stage

(b) fine-tuning stage

Figure 1: Framework for the agent with two steps optimization in vision-and-language navigation task.

tion learning by utilizing expert trajectories through behaviour cloning (Pomerleau, 1991; Fried et al., 2018). However, behaviour cloning was prone to the out-of-distribution trajectory once it was applied into the environment. Previous approach used the adversarial inverse reinforcement learning (AIRL) (Fu et al., 2018) which defined the reward function based on the expert trajectories (Zhou and Small, 2021) and used the learned reward function to train the agent through interactions with the environment. Other works developed the cross-modality matching (Wang et al., 2019) and model-based RL (Wang et al., 2018) to improve RL agent performance. Although previous methods have shown promising results, all of them required the exploration to the unseen environment to obtain additional training data when being evaluated in the unseen validation set. This scenario clearly did not represent real-world implementation where robot needed to provide appropriate actions without requiring any explorations. Therefore, in this work, the variational state tracking is proposed to improve generalization. Therefore, the agent can perform properly in unseen environments without requiring any environment exploration.

4 Attentive State Tracker and Navigator

4.1 Framework overview

Figure 1 illustrates the framework of agent in VLN task. The process of learning can be divided into two stages, the pre-training (Figure 1(a)) and the fine-tuning stages (Figure 1(b)). Meanwhile, the common setup of VLN agent consists of three main components including state tracker, agent policy, and recurrent experience replay. The state tracker involves an observation encoder, a summarization module, and a tracking module. The observation encoder takes the inputs of instruction $\mathbf{U}$, vision $\mathbf{V}$, and pose information $\mathbf{p}$ to extract the observation features $\mathbf{o}$. The summarization module is constructed according to an attention mechanism to summarize the given instruction to the meaningful representations for the agent. Then, the agent will pay more attention to the components of instruction which have higher attention score. Lastly, the tracking module can be implemented in either deterministic or stochastic way.

This paper presents two kinds of state trackers, deterministic and stochastic tracking module which are named as the attentive state tracker (AST) and the attentive variational state tracker (AVAST), respectively. AST is similar to the state tracker used in some of the prior works (Fried et al., 2018; Wang et al., 2019; Zhou and Small, 2021). Meanwhile, AVAST is a new state tracker that is proposed in this work. In a common VLN setup, an agent can be designed either using sequence-to-sequence (Seq2Seq) or RL agent by fine-tuning the Seq2Seq model through interactions with the environment. As shown in the figure, a Seq2Seq agent will be used in the pre-training stage based on the behavior cloning to provide stable state tracker which will carry out a stationary state representation. Meanwhile, in the fine-tuning stage, REINFORCE (Williams, 1992) is implemented to improve the performance. Due to POMDP property in VLN task, the transition information $\{\mathbf{o}_t, \mathbf{a}_t, r_t\}$ stored in the experience replay is dependent on the previous trajectories because of the incomplete information provided by the environment. Therefore, a recurrent experience replay is used to replace standard experience replay which was commonly used in MDP task.

4.2 Observation encoder

Both AST and AVAST involve an observation encoder that will extract meaningful features from $[\mathbf{U}; \mathbf{V}_t; \mathbf{p}_t]$. The natural language instruction matrix is denoted as $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_l, \ldots, \mathbf{u}_L]^\top$, where $\mathbf{u}_l$ is a word embedding from GloVe (Pennington et al., 2014) to represent the l-th word in the instruction and L is the length of the instruction. We feed the instruction matrix $\mathbf{U}$ into a recurrent model $f_u(\cdot)$ to obtain the initial context $\mathbf{H}^u = [\mathbf{h}_1^u, \ldots, \mathbf{h}_l^u, \ldots, \mathbf{h}_L^u]^\top$, and send the last hidden feature state $\mathbf{h}_L^u$ into a fully-connected network $g_\mathcal{T}(\cdot)$ to capture the initial trajectory information $\mathbf{h}_0^\mathcal{T}$ as follows

$$\begin{aligned} \mathbf{h}_1^u &= f_u\left(\mathbf{u}_1, \mathbf{h}_0^u\right) \\ &\vdots \\ \mathbf{h}_L^u &= f_u\left(\mathbf{u}_L, \mathbf{h}_{L-1}^u\right) \\ \mathbf{h}_0^\mathcal{T} &= g_\mathcal{T}\left(\mathbf{h}_L^u\right). \end{aligned} \tag{1}$$

The panoramic vision $\mathbf{V}_t$ is a representation of 36 first-person camera view images at time step t, and it is denoted as $\mathbf{V}_t = [\mathbf{v}_{t,1}, \ldots, \mathbf{v}_{t,i}, \ldots, \mathbf{v}_{t,36}]^\top$, where $\mathbf{v}_{t,i}$ is a vision feature to represent the i-th camera view image at time step t. The vision feature $\mathbf{v}_{t,i} = [\mathbf{v}_{t,i}^{\text{ResNet}}; \mathbf{v}_{t,i}^{\text{Orientation}}]$ is a concatenation of an image feature $\mathbf{v}_{t,i}^{\text{ResNet}}$ and an orientation feature $\mathbf{v}_{t,i}^{\text{Orientation}}$. An image feature $\mathbf{v}_{t,i}^{\text{ResNet}}$ is a 2048-dimensional vector extracted from a pre-trained ResNet-152 model (He et al., 2016), and an orientation feature is a 128-dimensional vector that repeats $[\sin\alpha_{t,i}, \cos\alpha_{t,i}, \sin\beta_{t,i}, \cos\beta_{t,i}]$ 32 times representing environmental views where $\alpha_{t,i}$ and $\beta_{t,i}$ are the relevant heading and elevation to the current camera pose, respectively. The vision embedding $\mathbf{e}_t^v$ is extracted by a cross-attention (Vaswani et al., 2017) module. This paper uses trajectory information $\mathbf{h}_{t-1}^\mathcal{T}$ from the state tracker as a query to attend the panoramic vision $\mathbf{V}_t$ using parameters $\{\mathbf{W}_v^q, \mathbf{W}_v^k\}$ via

$$\mathbf{e}_t^v = f_v\left(\mathbf{V}_t, \mathbf{h}_{t-1}^\mathcal{T}\right) = \left(\text{Softmax}(\mathbf{q}_v^\top \mathbf{K}_v) \cdot \mathbf{V}_t\right)^\top \tag{2}$$

where $\mathbf{q}_v = \mathbf{h}_{t-1}^\mathcal{T}\mathbf{W}_v^q$, $\mathbf{K}_v = \mathbf{V}_t\mathbf{W}_v^k$. The pose information $\mathbf{p}_t$ represents the current camera pose, and it is an 128-dimensional vector that repeats $[\sin\alpha_t, \cos\alpha_t, \sin\beta_t, \cos\beta_t]$ 32 times. α_t and β_t are the absolute heading and absolute elevation of the agent. To calculate the pose embedding $\mathbf{e}_t^p$, we feed the pose information $\mathbf{p}_t$ into a fully connected network $f_p(\cdot)$ in a form of

$$\mathbf{e}_t^p = f_p\left(\mathbf{p}_t\right). \tag{3}$$

4.3 Attentive variational state tracker

After the raw features $[\mathbf{U}; \mathbf{V}_t; \mathbf{p}_t]$ are encoded into $[\mathbf{H}^u; \mathbf{e}_t^v; \mathbf{e}_t^p]$, these encoded features are fed into the tracker, which is constructed by an *attentive summarization* module for instructions $\mathbf{H}^u$ and a stochastic tracking module for vision and pose information $[\mathbf{e}_t^v; \mathbf{e}_t^p]$. The tracker will generate the belief state $\mathbf{s}_t = [\mathbf{s}_t^u; \mathbf{s}_t^\mathcal{T}]$ and the trajectory information $\mathbf{h}_t^\mathcal{T}$ at each time step t. The attentive summarization module aims to summarize the instruction from initial context $\mathbf{H}^u$ into context belief state $\mathbf{s}_t^u$ to inform which words should the agent pay more attention. Next, the agent takes the context belief state $\mathbf{s}_t^u$ as a part of consideration to predict the action $\mathbf{a}_t$ at each time step t. In order to do so, the summarization module is constructed based on the attention mechanism. The trajectory information $\mathbf{h}_t^\mathcal{T}$ can be used as the query to attend over the instruction $\mathbf{H}^u$, and the word representation $\mathbf{h}_l^u$ can be weighted by the attention weight. Then, the weighted sum is treated as the context belief state $\mathbf{s}_t^u$. The procedure for generating the context belief state can be formulated using parameters $\{\mathbf{W}_u^q, \mathbf{W}_u^k, \mathbf{W}_u^v\}$ via

$$\mathbf{s}_t^u = g_u\left(\mathbf{H}^u, \mathbf{h}_t^\mathcal{T}\right) = \left(\text{Softmax}(\mathbf{q}_u^\top \mathbf{K}_u) \cdot \mathbf{V}_u\right)^\top \tag{4}$$

where $\mathbf{q}_u = \mathbf{h}_t^\mathcal{T}\mathbf{W}_u^q$, $\mathbf{K}_u = \mathbf{H}^u\mathbf{W}_u^k$, $\mathbf{V}_u = \mathbf{H}^u\mathbf{W}_u^v$. Considering the sufficient information process (Striebel, 1965), the belief state $\mathbf{s}_t$ is estimated based on the complete information state I_t^C. In VLN task, the observation $\mathbf{o}_t$ can be divided into, instruction $\mathbf{U}$, vision $\mathbf{V}_t$, and pose information $\mathbf{p}_t$, and the previous action information $\mathbf{a}_{t-1}$ can be implied by the current pose information $\mathbf{p}_t$. So, the complete information state I_t^C in VLN can be reshaped as follows

$$I_t^C = \langle \rho(\mathbf{s}_0), \mathbf{U}, \mathbf{V}_1, \mathbf{p}_1, \mathbf{V}_2, \mathbf{p}_2, \ldots, \mathbf{V}_t, \mathbf{p}_t \rangle. \tag{5}$$

The tracking module aims to generate the tracking belief state $\mathbf{s}_t^\mathcal{T}$ based on the complete information state I_t^C. Referring to some prior methods (Hausknecht and Stone, 2015; Lee et al., 2020), approaches for generating tracking belief state can be divided into two main methods, aggregation and estimation. In this work, we build two kinds of tracking model by using deterministic aggregation and stochastic estimation, which can be constructed by LSTM and Variationl Recurrent Neural Network (VRNN) (Chung et al., 2015) respectively. Both methods equip an aggregation module $g_\mathcal{T}$ to encode the history into trajectory information $\mathbf{h}_t^\mathcal{T}$ to represent the complete information state I_t^C. The aggregation modules can be generally expressed as

$$\mathbf{h}_t^\mathcal{T} = \begin{cases} g_{\mathcal{T}_0}\left(\mathbf{h}_L^u\right) & t = 0 \\ g_\mathcal{T}\left(\mathbf{o}_{\le t}\right) & t > 0 \end{cases} \tag{6}$$

where $\mathbf{o}_{\le t} = \{\mathbf{o}_1, \ldots, \mathbf{o}_t\}$.

The tracking module constructed by LSTM is a straightforward and deterministic method to aggregate the history information. This method has also been proposed to address POMDP problem (Hausknecht and Stone, 2015). $g_{\mathcal{T}}^{\text{LSTM}}$ denotes aggregation module $g_{\mathcal{T}}$ constructed by a LSTM model, and it is denoted as. The LSTM tracking module will directly treat the hidden feature state $\mathbf{h}_t^{\mathcal{T}}$ from the aggregation module as the belief state $\mathbf{s}_t^{\mathcal{T}}$. In the implementation, the initial hidden and cell feature-state of the LSTM aggregation module $g_{\mathcal{T}}^{\text{LSTM}}$ are both initialized from the last hidden and cell feature-state of the instruction LSTM encoder f_u to memorize the guided information. The procedure of generating a tracking belief state based on LSTM is formulated by

$$\mathbf{h}_t^{\mathcal{T}} = \begin{cases} g_{\mathcal{T}0}\left(\mathbf{h}_L^u\right) & t = 0 \\ g_{\mathcal{T}}^{\text{LSTM}}\left(\left[\mathbf{e}_t^v; \mathbf{e}_t^p\right], \mathbf{h}_{t-1}^{\mathcal{T}}\right) & t > 0 \end{cases} \quad (7)$$
$$\mathbf{s}_t^{\mathcal{T}} = \mathbf{h}_t^{\mathcal{T}}.$$

In order to improve model generalization, we propose the stochastic version of tracking module which is constructed by using VRNN. It will estimate the distribution $p(\mathbf{s}_t^{\mathcal{T}}|I_t^C)$ which will be sampled in every turn. Same as the original VRNN (Chung et al., 2015), there also exists an aggregation module $g_{\mathcal{T}}^{\text{VRNN}}$ to encode the trajectory information in this tracking module. Similar to the LSTM tracking module $g_{\mathcal{T}}^{\text{LSTM}}$, the embedding of the last hidden feature-state $\mathbf{h}_L^u$ from the instruction LSTM encoder f_u is used to be the initial trajectory information $\mathbf{h}_0^{\mathcal{T}} = g_{\mathcal{T}0}\left(\mathbf{h}_L^u\right)$ for the aggregation module. However, the input of $g_{\mathcal{T}}^{\text{VRNN}}$ is different from $g_{\mathcal{T}}^{\text{LSTM}}$. The input of $g_{\mathcal{T}}^{\text{VRNN}}$ includes not only the vision $\mathbf{e}_t^v$ and pose information $\mathbf{e}_t^p$ but also the tracking belief state $\mathbf{s}_t^{\mathcal{T}}$ to record the latent variable, sampled from the tracking belief state distribution. Identical to the LSTM tracking module, the complete information state I_t^C can be represented as the trajectory information $\mathbf{h}_t^{\mathcal{T}}$. The aggregation module in VRNN (Chien and Wang, 2022; Chien et al., 2017; Chien and Tsai, 2021) is also constructed by a LSTM model and can be expressed by

$$\mathbf{h}_t^{\mathcal{T}} = \begin{cases} g_{\mathcal{T}0}\left(\mathbf{h}_L^u\right) & t = 0 \\ g_{\mathcal{T}}^{\text{VRNN}}\left(\left[\mathbf{e}_t^v; \mathbf{e}_t^p; \mathbf{s}_t^{\mathcal{T}}\right], \mathbf{h}_{t-1}^{\mathcal{T}}\right) & t > 0. \end{cases} \quad (8)$$

To allow the sampling of tracking belief state $\mathbf{s}_t^{\mathcal{T}}$ at each time step t, VRNN aims to approximate the belief state distribution. The variational inference will sample a current belief state $\mathbf{s}_t^{\mathcal{T}}$ from the posterior based on the current observation and previous trajectory information $\mathbf{h}_{t-1}^{\mathcal{T}}$ from the aggregation model $g_{\mathcal{T}}$. Furthermore, we also need to build a prior distribution and conditional likelihood to reconstruct the observation for the self-learning criterion as shown in Eq. (16). The calculations of prior, posterior and likelihood using this VRNN are yielded by

$$\text{prior:} p(\mathbf{s}_t^{\mathcal{T}}|\mathbf{o}_{<t}, \mathbf{s}_{<t}^{\mathcal{T}}) = p(\mathbf{s}_t^{\mathcal{T}}|\mathbf{h}_{t-1}^{\mathcal{T}}) \quad (9)$$
$$\text{post:} q(\mathbf{s}_t^{\mathcal{T}}|\mathbf{o}_{\leq t}, \mathbf{s}_{<t}^{\mathcal{T}}) = q(\mathbf{s}_t^{\mathcal{T}}|\left[\mathbf{e}_t^v, \mathbf{e}_t^p\right], \mathbf{h}_{t-1}^{\mathcal{T}}) \quad (10)$$
$$\text{likel:} p(\mathbf{o}_t|\mathbf{s}_{\leq t}^{\mathcal{T}}, \mathbf{o}_{<t}) = p(\mathbf{v}_{t,\hat{i}}|\mathbf{s}_t^{\mathcal{T}}, \mathbf{h}_{t-1}^{\mathcal{T}}) \quad (11)$$

where $\mathbf{v}_{t,\hat{i}} = [\mathbf{v}_{t,\hat{i}}^{\text{ResNet}}; \mathbf{v}_{t,\hat{i}}^{\text{Orientation}}]$ is the intention vision embedding. Agent will change its current perspective from i to $\hat{i}$ before it moves to the next position at each time step. To provide stationary state representation, both AST and AVAST will be pre-trained based on a Seq2Seq agent. AST can be constructed with an attentive summarization module, a tracking module constructed by LSTM, and the observation encoders mentioned previously. The objective of AST pre-training is shown by

$$\mathcal{J}_\pi = \mathbb{E}_{(\mathbf{o}_t, \mathbf{a}_t^\star)\sim D}\left[\pi\left(\mathbf{a}_t^\star|\left[\mathbf{s}_t^u; \mathbf{s}_t^{\mathcal{T}}\right]\right)\right] \quad (12)$$

where

$$\mathbf{s}_t^{\mathcal{T}} = \mathbf{h}_t^{\mathcal{T}} = g_{\mathcal{T}}^{\text{LSTM}}(\mathbf{o}_t, \mathbf{h}_{t-1}^{\mathcal{T}}). \quad (13)$$

Different from AST, AVAST replaces the LSTM tracking module in AST with a variational tracking module using VRNN (Chien and Wang, 2019). The objective $\mathcal{J}_\pi$ for pre-training AVAST can be expressed in a form of

$$\mathbb{E}_{(\mathbf{o}_t, \mathbf{a}_t^\star)\sim D}\left[\mathbb{E}_{\mathbf{s}_t^{\mathcal{T}}\sim q\left(\mathbf{s}_t^{\mathcal{T}}|\mathbf{o}_t, \mathbf{h}_{t-1}^{\mathcal{T}}\right)}\left[\pi\left(\mathbf{a}_t^\star|\left[\mathbf{s}_t^u; \mathbf{s}_t^{\mathcal{T}}\right]\right)\right]\right] \quad (14)$$

using

$$\mathbf{h}_{t-1}^{\mathcal{T}} = g_{\mathcal{T}}^{\text{VRNN}}\left(\left[\mathbf{o}_{t-1}; \mathbf{s}_{t-1}^{\mathcal{T}}\right], \mathbf{h}_{t-2}^{\mathcal{T}}\right). \quad (15)$$

Rather than learning the signal which only depends on the downstream task for the LSTM tracking module, VRNN has an additional learning signal to jointly enhance the performance for the tracking belief state representation. The evidence lower bound $\mathcal{J}_{\text{ELBO}}$ can be derived as shown in Eq. (16) to be the additional learning criterion for VRNN

Algorithm 1: Pre-training state tracker

```
Preprocess R2R dataset $D$
Initialize state tracker parameters $\psi$
Initialize Seq2Seq agent parameters $\phi$
while not converged do
    for each $\{\mathbf{U}, \mathbf{V}_{1:T}, \mathbf{p}_{1:T}, \mathbf{a}^{\star}_{1:T}\} \in D$ do
        get $\mathbf{H}^{u}$ based on Eq. (1)
        get $\mathbf{e}^{v}_{1:T}, \mathbf{e}^{p}_{1:T}$ based on Eqs. (2)(3)
        get $\mathbf{s}^{u}_{1:T}$ based on Eq. (4)
        if state tracker is AVAST then
            get $\mathbf{s}^{\tau}_{1:T}$ based on Eqs. (8)(11)
            update $\psi, \phi$ based on
              Eqs. (14)(16)
        end
        if state tracker is AST then
            get $\mathbf{s}^{\tau}_{1:T}$ based on Eq. (7)
            update $\psi, \phi$ based on Eq. (12)
        end
    end
end
```

via

$$
\begin{aligned}
\ln p(\mathbf{o}_{\leq T}) &= \ln \int p(\mathbf{o}_{\leq T}, \mathbf{s}^{\tau}_{\leq T}) \\
&\geq \mathbb{E}_{q(\mathbf{s}^{\tau}_{\leq T} | \mathbf{o}_{\leq T})} \left[\ln \frac{p(\mathbf{o}_{\leq T}, \mathbf{s}^{\tau}_{\leq T})}{q(\mathbf{s}^{\tau}_{\leq T} | \mathbf{o}_{\leq T})} \right] \\
&= \mathbb{E}_{q(\mathbf{s}^{\tau}_{\leq T} | \mathbf{o}_{\leq T})} \Bigg[\sum_{t=1}^{T} \ln p(\mathbf{o}_t | \mathbf{s}^{\tau}_{\leq T}, \mathbf{o}_{<t}) \\
&\quad - D_{\mathrm{KL}} \left(p \left(\mathbf{s}^{\tau}_t | \mathbf{o}^{\tau}_{<t}, \mathbf{s}^{\tau}_{<t} \right) \| q \left(\mathbf{s}^{\tau}_t | \mathbf{o}_{\leq t}, \mathbf{s}^{\tau}_{<t} \right) \right) \Bigg] \\
&= \mathcal{J}_{\mathrm{ELBO}}.
\end{aligned}
\tag{16}
$$

Pre-training procedure of AST and AVAST based on a Seq2Seq agent can be seen in Algorithm 1.

5 Experiments

5.1 Experimental setup

The proposed method was evaluated in VLN task using room-to-room (R2R) dataset, which contains pairs of path and instruction based on human annotation with Matterport3D simulator. It is built based on Matterport3D dataset (Chang et al., 2017), which is a large RGB-D dataset of building-scale scenes. In order to meet the real-world situation, the agent should be prevented from crossing the wall and floor or jumping to a non-navigable place. The action space in the simulator is based on a pre-defined undirected graph over panoramic viewpoints, $\mathcal{G} = \langle \mathcal{P}, \mathcal{E} \rangle$. The agent's actions are limited in a way that they can only navigate to the viewpoint, which is adjacent to the current viewpoint based on the graph $\mathcal{G}$. At each time step t, agent is provided with next-step navigable viewpoints set $\mathcal{A}_t$ in a form of

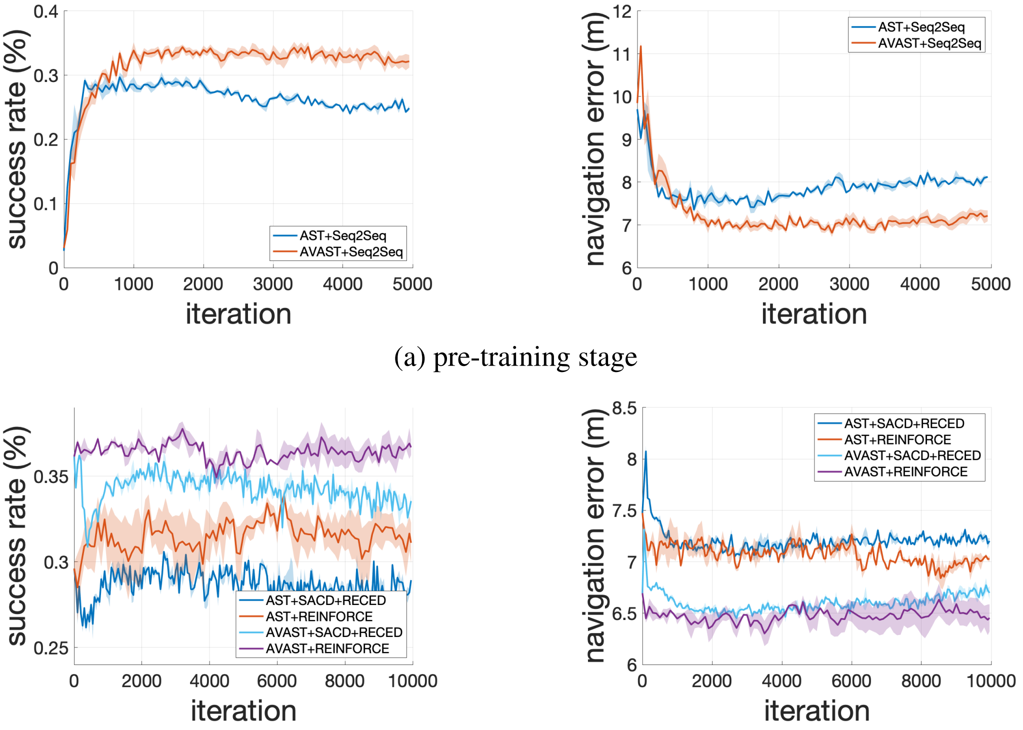
$$
\mathcal{A}_t = \{\rho_t\} \cup \{\rho_i \in \mathcal{P} | \langle \rho_i, \rho_j \rangle \in \mathcal{E} \wedge \rho_i \in \mathcal{R}_t\} \tag{17}
$$

where ρ_t is the current viewpoint and $\mathcal{R}_t$ is the region of space enclosed by the left and right extents of the camera view frustum at step t. The simulator only define the navigable set $\mathcal{A}_t$ to the current viewpoint ρ_t and handles how to update next viewpoint ρ_{t+1}, camera heading α, and camera elevation β after next viewpoint ρ_{t+1} is selected by the agent to navigate. Although the simplified discrete simulator provides a clear problem formulation, this kind of low-level control interface is non-trivial to be applied for training a navigation agent. Moreover, following the original approach (Anderson et al., 2018), the simulator needs to aggregate two possible ways to generate the visual observation, from the raw RGB image and pre-trained ResNet embedding to represent the current vision observation. This procedure makes the simulator to be dependant on the huge Matterport3D dataset and requires a complicated setup procedure.

Due to the aforementioned reasons, we build a simpler VLN environment that is not dependant on Matterport3D dataset and can be relatively easier to set up a simulation. Similar to the previous approaches (Fried et al., 2018; Zhou and Small, 2021), the proposed VLN environment provides a panoramic interface with discrete control for navigation agents. The action space is different from the original Matterport3D simulator in Eq. (17) in a way of

$$
\mathcal{A}_t = \{\rho_t\} \cup \{\rho_i \in \mathcal{P} | \langle \rho_i, \rho_j \rangle \in \mathcal{E}\}. \tag{18}
$$

As a result, the agent can navigate to a nearby viewpoint, without any need to be enclosed by the left and right extents of the camera view frustum at step t. Furthermore, we directly build a mapping table to look up the desired ResNet embedding $\mathbf{V}_t$ at each time step t to eliminate the dependancy on Matterport3D dataset. During setup, the VLN environment will initialize the word embedding from GloVe (Pennington et al., 2014) to transform natural language instructions $x = [x_1, \ldots, x_l, \ldots, x_L]$ into instruction matrices



(a) pre-training stage

(b) fine-tuning stage

Figure 2: Comparison of the results in unseen validation during pre-training and fine-tuning phases. Both pre-training and fine-tuning experiments do not truncate the instructions or use the augmented data from Speaker-Follower. The mean curve and standard deviation region are drawn by running the same experiment in multiple random seeds.

$\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_l, \ldots, \mathbf{u}_L]$ (Watanabe and Chien, 2015). The interface of VLN environment is designed to be closer to the typical RL environment, Gym. At the beginning of each episode, VLN environment provides instruction matrix $\mathbf{U}$, vision observation $\mathbf{V}_1$, pose information $\mathbf{p}_1$, and navigable viewpoint set $\mathcal{A}_1$. After the agent act an action $\mathbf{a}_t$, VLN environment will generate the next vision observation $\mathbf{V}_{t+1}$, pose information $\mathbf{p}_{t+1}$, navigable viewpoint set $\mathcal{A}_{t+1}$, and reward r_t. Reward r_t are defined as follows:

$$r_t = \begin{cases} D(\rho_{t-1}, \rho_{\text{goal}}) - D(\rho_t, \rho_{\text{goal}}) & t < T \\ \mathbf{1}\left[D(\rho_t, \rho_{\text{goal}}) \leq 3\right] & t = T \end{cases} \tag{19}$$

where $D(\rho_i, \rho_j)$ denotes the shortest path distance between locations ρ_i and ρ_j, and ρ_{goal} denotes the location of goal. For the evaluation metrics, this paper consider two metrics which are navigation error (NE) and success rate (SR). NE measures the shortest path between the goal location and final location of the agent's path. SR measures the average rate of the agent stopping within 3 meters near to the goal location.

5.2 Experimental results

In order to evaluate the effectiveness of AVAST, we highly focus on the unseen validation task, because it represents more real-world scenario where the agent frequently faces unseen environment during implementation. To provide stationary state representation for RL agent, both AST and AVAST were initially trained based on Seq2Seq agent via behaviour cloning algorithm. The learning curves are shown in Figure 2(a) where AVAST convincingly outperformed AST indicated by higher success rate and lower navigation turn over iterations. Next, both AST+Seq2Seq and AVAST+Seq2Seq performances were compared to the prior baseline methods, which are Speaker-Follower (SF) (Fried et al., 2018) and Inverse Reinforcement Learning with Natural Language Goals (LangGoalIRL) (Zhou and Small, 2021). The performance of the proposed method and baseline methods are shown in Table 1. Based on the result, the generalization improvement could be achieved by using AVAST, indicated by the lowest navigation error and the highest success rate compared to the baselines with convincing performance gap.

#	Model	NE ↓	SR ↑
1	SF†*	7.07	31.2
2	LangGoalIRL†*	-	30.0
3	AST + Seq2Seq†*	7.54	29.1
4	AVAST + Seq2Seq†*	**6.60**	**36.6**

Table 1: Navigation errors (NE) and success rates (SR) for different behavior cloning methods in VLN unseen validation datasets. (†: trained without using augmented data. *: trained based on pure behavior cloning).

To enhance the agent performance further, the model was fine-tuned using REINFORCE algorithm (Williams, 1992). In this fine-tuning evaluation, two previous approaches were introduced to be the experiment baselines. The first is discrete version of soft actor critic (SACD) (Christodoulou, 2019; Chien and Yang, 2021) which has shown improvement in the LangGoalIRL. The second is the curriculum learning with the recurrent replay distributed DQN from demonstrations (R2D3) (Paine et al., 2020) which we name it as recurrent experience replay with curriculum expert demonstrations (RECED). The learning curves of fine-tuning process are shown in Figure 2(b). Meanwhile, the final evaluation result can be seen in Table 2. In the last evaluation, an additional baseline, reinforced cross-modal matching (RCM) (Wang et al., 2019) which involved instruction truncation to improve the performance is introduced. Although this trick can improve learning efficiency, it is not really fit to the real-world scenario. Accordingly, in our main experiments in Table 1 and Table 2, we did not truncate natural language instructions into a certain length. However, in order to show the generalization of AVAST, the experiments under same setting with RCM was conducted, and the results are shown in Table 3. Based on these results, there are four findings which are summarized as follows.

1. **Variational state tracker provided better generalization in unseen validation.** From the learning curve as shown in Figure 2(a), we can notice that agent performed better than the one using AST as a state tracker without suffering overfitting issue due to the ability of AVAST in providing more general state representation in unseen validation. Furthermore, as shown in Table 1, AVAST+Seq2Seq outperformed the methods which were purely trained via behavior cloning algorithm.

#	Model	NE ↓	SR ↑
1	SF*	6.62	35.5
2	LangGoalIRL†	-	30.8
3	LangGoalIRL	-	35.7
4	AST + SACD + RECED†	7.06	31.3
5	AST + REINFORCE†	6.92	34.4
6	AVAST + SACD + RECED†	6.44	36.7
7	AVAST + REINFORCE†	**6.22**	**38.5**

Table 2: Navigation errors and success rates for different methods in VLN unseen validation datasets (†: trained without using augmented data; *: trained based on pure behavior cloning).

#	Model	NE ↓	SR ↑
1	SF*	6.62	35.5
2	RCM‡	6.02	40.6
3	LangGoalIRL	-	35.7
4	AVAST + REINFORCE	**6.01**	**42.2**

Table 3: Navigation errors and success rates for different methods in VLN unseen validation datasets under the scenario of truncating instruction (*: trained based on behavior cloning, ‡: trained without intrinsic rewards).

2. **Agent's performance was improved via fine-tuning based on RL algorithms, leading to outperforming the baseline methods.** We can notice from Table 2, after fine-tuning the pre-trained model, AVAST+REINFORCE performed better compared to the other baseline methods in unseen validation. This result indicates that the model has successfully taken advantage of exploration property in the REINFORCE algorithm.

3. **Introducing expert could not improve the agent performance.** As it can be seen from Figure 2(b), the performance of both AVAST and AST trained with expert demonstrations in a progressive way did not improve the performance. Instead, it degraded the agent performance compared to those that were trained with REINFORCE algorithm. This result indicates that the distribution of the unseen environment is quite different compared to the training environment.

4. **Hard exploration issue led to poor state-action value estimation for policy to learn.** We can notice from Figure 2(b), the curves of

both AVAST and AST with SACD dropped in the beginning due to poor value estimation from the critic network. Once the critic network was unable to provide a precise value estimation, the policy would be led to a bad direction, resulting in harmed performance.

6 Conclusions

This paper has presented attentive variational state tracker to deal with the generalization issue in vision-and-language navigation task. This method developed a variational approach to fulfill the partially observable Markov decision process where the belief states were sampled to implement the stochastic machine to improve the generalization to unseen environments. The experimental results demonstrated that the policy optimization using REINFORCE in combination of the proposed AVAST outperformed the previous methods in terms of navigation errors and success rates. The generalization was assured by the evaluation in the unseen environments.

Acknowledgement

This work was supported in part by the National Science and Technology Council, Taiwan, under contract MOST 110-2634-F-A49-003.

References

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3674–3683.

Karl Johan Åström. 1965. Optimal control of markov processes with incomplete state information. *Journal of Mathematical Analysis and Applications*.

Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. 2013. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*.

Angel X. Chang, Angela Dai, Thomas A. Funkhouser, Maciej Halber, Matthias Nießner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3D: Learning from RGB-D data in indoor environments. In *Proc. of International Conference on 3D Vision*, pages 667–676.

Jen-Tzung Chien, Wei-Lin Liao, and Issam El Naqa. 2021. Exploring state transition uncertainty in variational reinforcement learning. In *Proc. of European Signal Processing Conference*, pages 1527–1531.

Jen-Tzung Chien, Chen Shen, et al. 2017. Stochastic recurrent neural network for speech recognition. In *Proc. of Annual Conference of International Speech Communication Association*, pages 1313–1317.

Jen-Tzung Chien and Chih-Jung Tsai. 2021. Variational sequential modeling, learning and understanding. In *Proc. of IEEE Automatic Speech Recognition and Understanding Workshop*, pages 480–486.

Jen-Tzung Chien and Chun-Wei Wang. 2019. Self attention in variational sequential learning for summarization. In *Proc. of Annual Conference of International Speech Communication Association*, pages 1318–1322.

Jen-Tzung Chien and Chun-Wei Wang. 2022. Hierarchical and self-attended sequence autoencoder. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):4975–4986.

Jen-Tzung Chien and Shu-Hsiang Yang. 2021. Model-based soft actor-critic. In *Proc. of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, pages 2028–2035.

Petros Christodoulou. 2019. Soft actor-critic for discrete action settings. *arXiv preprint arXiv:1910.07207*.

Chang-Ting Chu, Mahdin Rohmatillah, Ching-Hsien Lee, and Jen-Tzung Chien. 2022. Augmentation strategy optimization for language understanding. In *Proc. of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7952–7956.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. *Advances in Neural Information Processing Systems*, 28:2980–2988.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. In *Advances in Neural Information Processing Systems*, pages 3318–3329.

Justin Fu, Katie Luo, and Sergey Levine. 2018. Learning robust rewards with adversarial inverse reinforcement learning. In *Proc. of International Conference on Learning Representations*.

Matthew J. Hausknecht and Peter Stone. 2015. Deep recurrent Q-learning for partially observable MDPs. In *Proc. of Association for the Advancement of Artificial Intelligence*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–778.

Chuan-En Hsu, Mahdin Rohmatillah, and Jen-Tzung Chien. 2021. Multitask generative adversarial imitation learning for multi-domain dialogue system. In *Proc. of IEEE Automatic Speech Recognition and Understanding Workshop*, pages 954–961.

Maximilian Igl, Luisa M. Zintgraf, Tuan Anh Le, Frank Wood, and Shimon Whiteson. 2018. Deep variational reinforcement learning for pomdps. In *Proc. of International Conference on Machine Learning*.

Youngsoo Jang, Jongmin Lee, and Kee-Eung Kim. 2022. GPT-critic: Offline reinforcement learning for end-to-end task-oriented dialogue systems. In *International Conference on Learning Representations*.

Michal Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaskowski. 2016. Vizdoom: A doom-based AI research platform for visual reinforcement learning. In *Proc. of IEEE Conference on Computational Intelligence and Games*.

Alex Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. 2020. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *Advances in Neural Information Processing Systems*, 33.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature*.

Tom Le Paine, Caglar Gulcehre, Bobak Shahriari, Misha Denil, Matt Hoffman, Hubert Soyer, Richard Tanburn, Steven Kapturowski, Neil Rabinowitz, Duncan Williams, et al. 2020. Making efficient use of demonstrations to solve hard exploration problems. In *Proc. of International Conference on Learning Representations*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proc, of Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543.

Dean A. Pomerleau. 1991. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97.

Mahdin Rohmatillah and Jen-Tzung Chien. 2021a. Causal confusion reduction for robust multi-domain dialogue policy. In *Proc. of Annual Conference of International Speech Communication Association*, pages 3221–3225.

Mahdin Rohmatillah and Jen-Tzung Chien. 2021b. Corrective guidance and learning for dialogue management. In *Proc. of ACM International Conference on Information & Knowledge Management*, pages 1548–1557.

Charlotte Striebel. 1965. Sufficient statistics in the optimum control of stochastic systems. *Journal of Mathematical Analysis and Applications*, 12(3):576–592.

Emanuel Todorov, Tom Erez, and Yuval Tassa. 2012. MuJoCo: A physics engine for model-based control. In *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. 2019. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. 2018. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *Proc. of the European Conference on Computer Vision (ECCV)*.

Shinji Watanabe and Jen-Tzung Chien. 2015. *Bayesian speech and language processing*. Cambridge University Press.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, pages 229–256.

Li Zhou and Kevin Small. 2021. Inverse reinforcement learning with natural language goals. In *Proc. of Association for the Advancement of Artificial Intelligence*, pages 11116–11124.