Spatially-Aware Speaker for Vision-and-Language Navigation Instruction Generation

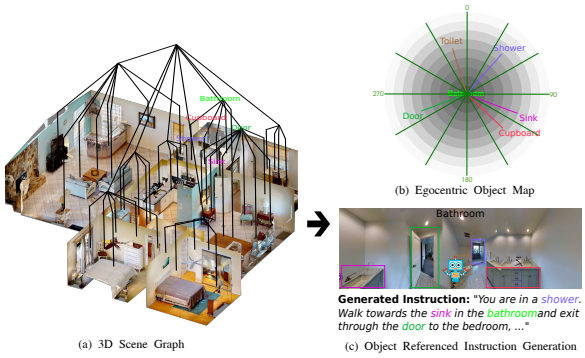
Muraleekrishna Gopinathan, Martin Masek, Jumana Abu-Khalaf, David Suter
Centre for Artificial Intelligence and Machine Learning
Edith Cowan University, 270 Joondalup Dr, Joondalup, WA 6027, Australia
{k.gopinathan,m.masek,j.abukhalaf,d.suter}@ecu.edu.au

Abstract

Embodied AI aims to develop robots that can *understand* and execute human language instructions, as well as communicate in natural languages. On this front, we study the task of generating highly detailed navigational instructions for the embodied robots to follow. Although recent studies have demonstrated significant leaps in the generation of step-by-step instructions from sequences of images, the generated instructions lack variety in terms of their referral to objects and landmarks. Existing speaker models learn strategies to evade the evaluation metrics and obtain higher scores even for low-quality sentences. In this work, we propose SAS (Spatially-Aware Speaker), an instruction generator or *Speaker* model that utilises both structural and semantic knowledge of the environment to produce richer instructions. For training, we employ a reward learning method in an adversarial setting to avoid systematic bias introduced by language evaluation metrics. Empirically, our method outperforms existing instruction generation models, evaluated using standard metrics. Our code is available at https://github.com/gmuraleekrishna/SAS.

1 Introduction

Incorporating language understanding in robots has been a long-standing goal of the NLP and robotic research community. Specifically, the Vision-Language Navigation (VLN) task requires robots to follow natural language instructions grounded on vision to navigate in human living spaces. Although humans generally follow navigational instructions well, training robots to follow natural language instructions remains a challenging problem. Detailed navigation instructions may include landmarks, actions, and destinations. Recent work has succeeded in improving instruction understanding of robots by augmenting instruction and trajectory training data (Hong et al., 2020; Wang et al., 2022, 2023). They showed that using machine-generated instructions from a large number of navigational paths sampled from real houses helps robots navigate successfully even in previously unseen environments. However, there is still room for improvement, as the quality of machine-generated instructions is clearly lower compared to human annotations (Zhao et al., 2021).



(a) 3D Scene Graph (b) Egocentric Object Map (c) Object Referenced Instruction Generation

Figure 1: Extracting 3D scene relationships from house environments (a,b) can improve instruction generation by including object references (c).

In this work, we present a novel instruction generation model that can produce a variety of human-like instructions using semantic and structural cues from the environment. Our method uses rooms, interesting landmarks, objects, inter-object relations, object locations, and spatial features to produce richer instructions that can be used by robots and humans alike (Fig. 1). Our Spatially-Aware Speaker (SAS) model generates information-rich instructions by leveraging expert demonstrations that map trajectories to verbal directions. Incorporating spatial references within these instructions is critical, as they convey the environmental layout, highlight key landmarks relevant to the actions taken, and gauge the progression of navigation. At its core, SAS employs a sequence learning framework that is fine-tuned through a combination of adversarial learning rewards and multiple objec-

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 13601–13614

tives aimed at enhancing its linguistic generation capabilities.

The architecture of SAS is based on an Encoder-Decoder model, which processes a sequence of viewpoints and corresponding actions that define a navigational path, subsequently generating a coherent set of instructions. During the encoding phase, the model extracts vital information from visual inputs, such as object categories (e.g., cupboard, bed), spatial relationships between objects (e.g., on top of, near, under), object placements, and significant landmarks within the viewpoint (e.g., bedroom, kitchen). These elements are combined with navigational actions to form a comprehensive vision-action representation that computes the temporal order.

The decoding phase acquires linguistic capabilities by linking this latent representation to the instructions encountered during training. An adversarial learning objective is introduced to encourage the generation of varied sentences, mitigating the potential biases that automatic evaluation metrics introduce. Through this novel approach, SAS outperforms existing instruction generation models on VLN datasets evaluated using standard language evaluation metrics.

Our contribution is as follows.

- We introduce a novel speaker model (SAS) that can incorporate semantic and structural viewpoint features into the instruction.
- We develop an adversarial reward learning strategy, that rewards diverse instructions, to train our SAS model.
- We introduce a large scale silver dataset for automatic data augmentation.

2 Related Work

2.1 Natural Language Navigational Guidance and Following

Methods for modelling human and robot behaviours for the generation and execution of natural language instructions span several disciplines, including cognitive psychology (Ward et al., 1986), sociology (Harrell et al., 2000), natural language processing (NLP) (Daniele et al., 2017), and robotics (Wang et al., 2023). Studies show that adequate navigational instructions have *directions*, *landmarks*, *region descriptions* and *turn-by-turn actions* (Look et al., 2005). These instructions are also beneficial to human-machine interaction, particularly in embodied agents.

The embodied navigation problem has been receiving attention from multiple research domains such as robotics, NLP and scene understanding (Anderson et al., 2018; Dorbala et al., 2023). Recent studies have shown that VLN agents learn better on machine-generated examples (Fried et al., 2018; Tan et al., 2019; Wang et al., 2023). These methods, generally called *Speaker models*, are still far from generating human-like instructions, as exhibited by their lower machine-generation evaluation scores. Our work aims to improve the quality of the generated instructions over baseline models by including landmarks, actions, and directions.

2.2 Spatial and Semantic Scene Understanding for Embodied Navigation

Neuroscience has shown that humans, like others in the animal kingdom, use spatial and temporal cues to build a cognitive map of their surroundings (Kuipers, 1978). These cognitive maps are crucial for manipulating or navigating the environment. Inspired by this, recent studies in robotics and embodied navigation have used vision models to infer the structure of the environment (Kuo et al., 2023; Gopinathan et al., 2021), spatial relationships among objects in the scene (Qi et al., 2020; Moudgil et al., 2021) and environment layout (Gopinathan et al., 2023) to learn about the environment (Song et al., 2022). While these methods utilise structure, spatial and semantic knowledge in various combinations to learn vision-language association, they are not applied to instruction generation task. In this work, we use all four aspects of a satisfactory instruction - *landmark* through visual encoding, *directions* through directional encoding, *turn-by-turn actions* using action encoding and *region descriptions* as semantic encoding - to generate richer instructions.

2.3 Reinforcement learning for Instruction Generation

Reinforcement learning (RL) has been successful in machine generation tasks such as translation, instruction generation, captioning, and storytelling. In this paper, the primary objective is to maximise the expected return of a word-generating policy. RL instruction generation methods are found to learn the target distribution better than traditional maximum likelihood estimation (MLE) algorithms due to the inherent *exposure bias* (Arora et al., 2022).

Applying RL to learn the target distribution requires extensive feature and reward engineering. Instead, inverse reinforcement learning (IRL) is proposed to infer the expert's reward function. Using an adversarial setting, IRL has been shown to improve visual story telling (Wang et al., 2018).

Existing work in VLN have studied natural language instruction generation (Fried et al., 2018; Wang et al., 2022, 2019) as a sequence generation problem, however, they focus on navigational success of the overall agent over instruction quality. Duo et al. Dou and Peng (2022) optimise their *Speaker* by using the similarity between itself and the gradient of the navigation model as a reward for RL. The authors evaluated their method on BLEU, a metric which does not guarantee high-quality instructions. Zhao et al. (2021) discovered that in the context of dialogue generation and navigational tasks, the majority of n-gram-based automatic language evaluation metrics show a weak correlation with human-annotated instructions.

Inspired by these studies, we adopt an IRL-based reward learning strategy to produce high-quality navigation instructions by indirectly learning the reward from language metrics. This mitigates exposure bias and avoids the model from gaming the metrics to achieve high evaluation scores - even with low-quality instructions.

3 Problem Definition

Here we present the task aimed at generating linguistic instructions from navigational demonstrations. The generation of instructions is posited as the converse operation to the standard VLN task, where an agent executes a navigation instruction in a house. For this, we develop a *Speaker* agent that synthesises a coherent set of natural language instructions from a sequence of navigational actions along a trajectory.

At each discrete time step t, the *Speaker* agent is presented with a panoramic visual observation O_t and a directional action a_t that signifies the transition to the next viewpoint within the trajectory. Upon completion of a navigational episode, the agent outputs the complete instruction of the traversed path, that is, $X = \{w_0, \ldots, w_l\}$. Formally, the objective of the *Speaker* is to minimise the negative log-likelihood for ground-truth instruction $\mathbf{X}$ conditioned on trajectory $T = \{O_1, \ldots, O_N\}$ with parameter θ:

$$L_\theta = -\sum_{\theta} \log(p(X|T;\theta))) \quad (1)$$

4 Preliminaries

4.1 Path Mixing

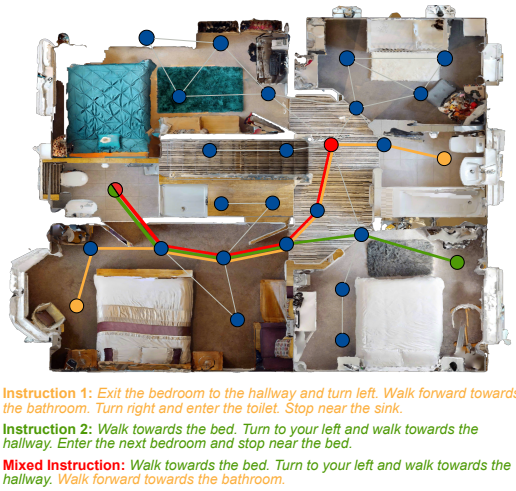


Figure 2: Path Mixing (PM) using fine-grained paths from R2R dataset. Original paths ●→● and ●→● are mixed to generate ●→●.

Prior work in VLN have shown that more instruction examples can improve an agent's performance in previously unseen environments (Moudgil et al., 2021; Wang et al., 2022). Hence, to supplement training data, we mix parts of trajectories from the FGR2R dataset (Hong et al., 2020) (which is derived from the R2R (Anderson et al., 2018) dataset) to obtain additional instruction-trajectory pairs.

R2R dataset contains paths and human-annotated instructions for navigating inside 3D scanned house environments from the Matterport3D (MP3D) dataset (Chang et al., 2017). Further fine-grained (turn-by-turn) or *micro-instructions* are available in the FGR2R dataset. We adopt FGR2R to enhance our training data by algorithmically combining parts of its trajectories, creating new instruction-trajectory pairs. Unlike REM (Moudgil et al., 2021; Liu et al., 2021), which randomly mixes data from different houses, we only mix trajectories from the same house to ensure visual and object referral consistency.

First, we identify key edges in the graph to mix trajectories, focusing on start edges ε_{start} and end edges ε_{end} of navigation paths. These edges are crucial as they relate to start and end of instructions (e.g. "Walk away from the desk, Turn right"). Edges with micro-instructions[1] that do not contain

[1]Part of the instruction pertinent to one edge of the path

a NOUN, VERB are ignored to avoid partial or inconsequential (*wait there*) actions. Then, we mix the remaining transition edges ε_{trans} to form a trajectory from ε_{start} to ε_{end}. Nodes that are too close to each other, leading to a lack of visual diversity or repetitive micro-instructions are also avoided. We connect edges based on the following criteria: (1) the distance between any two nodes (except the start and end nodes) should not exceed 3m, (2) the angles between edges should prevent looping; (3) the start and end nodes should not share an edge; (4) micro-instructions from shared edges of different trajectories are selected randomly.

The final instruction combines these micro-instructions, and the trajectory is a sequence of edges (as shown in Fig. 2). Using this method, we generated 162k instruction-trajectory pairs with path lengths ranging from 5m to 20m. The dataset[2] averages 7.27 views per trajectory, a mean path length of 14.4m, and about 82 words per instruction. More dataset statistics are added to the supplementary material.

4.2 Action Parsing

To allow the speaker to learn action phrases by associating them to the navigational actions, we automatically extract the action phrases from the instruction for training. For this, we use spaCy's (Honnibal et al., 2020) dependency parsing and part-of-speech tagging to identify verb forms that are transitive, indirect transitive, direct transitive and also the part-of-speech forms such as nouns, adverbs, adpositions, interjections and determiners (refer Appendix B for more details). We use this algorithm to identify action phrases from each step of the instruction and classify parts of sentences to either actions or other phrases. The identified micro-instructions are used to train the decoder part of SAS (§6.2).

5 Spatially-Aware Speaker (SAS) Model

In this section, we present our Spatially-Aware Speaker model. SAS is an encoder-decoder model which generates an instruction for a trajectory when the sequence of viewpoints and actions are provided. The *trajectory encoder* produces visual-action context from the trajectory viewpoints and the corresponding actions. This context is used by *instruction decoder* to generate instruction.

[2]Available at https://zenodo.org/records/10396782

To incorporate spatial and semantic awareness, we provide three crucial pieces of information to the model, namely: Action Encoding, Structural Encoding and Semantic Encoding. These are explained in the following sections.

Action Encoding The action taken between viewpoints in a trajectory is represented by Action Encoding. The visual action encoding is the current heading and elevation of the agent with respect to the next view direction. The relative elevation (θ) and heading (ϕ) angles are encoded as $E_a = [\cos\theta, \sin\theta, \cos\phi, \sin\phi]$.

Structural Encoding Structural encoding provides knowledge of the egocentric locations of objects with respect to the *Speaker*. In the panoramic view of each viewpoint, we extract an object's location in the image frame, as well as its size and distance to the agent in order to represent a complete pose of the object relative to the agent. The image frame location is obtained from the location of the object's bounding box detected by a Faster R-CNN (Ren et al., 2017) detector trained on the Visual Genome (Krishna et al., 2017) dataset. The size and distance of the object are obtained by projecting (inverse pinhole camera projection) the bounding box to the point cloud and measuring the centroid and volume of the contained point cloud. This provides an estimate of the size and distance to form the object descriptions. Effectively, the structural encoding is a combination of object features f_o, object location (c_x, cy), size s_o and distance d_o , respectively, i.e. $E_{so} = [f_o, (c_x, cy), s_o, d_o]$.

Semantic Encoding To provide inter-object relationships we use 1600 object classes Faster-RCNN and relationships extracted from ConceptNet (Speer et al., 2017). The object-to-object-room semantic features are a combination of GloVe embedding G of the respective token, $e^{obj}_{i,j,k}$= {G(<obj_i>);G(<rel_j>);G(<obj_k>)}.

Furthermore, we encode the relationship from room to object using the *in the* relation as $e^{room}_{l,m}$={G(<obj_l>);G(<in>);G(<$room_m$>)}. For each viewpoint, we encode one e_{room} and one e^{obj} per view direction with the highest detection confidence. In effect, we obtain the semantic encoding per viewpoint, $E_{sm} = \{(e^{obj}_1, e^{room}), \ldots, (e^{obj}_{36}, e^{room})\}$.

Panoramic Room-Object Attention Finally, we combine structural and semantic knowledge with panoramic view to obtain a panoramic knowledge

feature. For this we first concatenate the candidate feature f_c with the Structural Encoding E_{so} and Semantic Encoding E_{sm}:

$$f_{cs} = \mathbf{W}[f_c; E_{sm}; E_{so}] \quad (2)$$

where $\mathbf{W}$ is the trainable projection. We apply an attention module to make the model attend to information from different sub-spaces. The two projections Q(query) and K(key), which are from the action embedding and the candidate-semantic embedding, respectively, are applied to the attention as:

$$\alpha_k = \text{softmax}\left(\frac{Q(h_t^a)K^T(f_{cs})}{\sqrt{D_k}}\right) \quad (3)$$

$$c_k = \sum_{i,j} \alpha_{i,j} f_{p_{i,j}} \quad (4)$$

$$g_t = \tanh(\mathbf{W}[c_k; h_t^a]) \quad (5)$$

where c_α is the context vector, D_k is the hidden size of the attention layer, h_t^a is the hidden action state of the *trajectory encoder* and g_t is the gated output. The affinity matrix f_p governs the information flow between neighbouring view patches of the panorama. Finally, we capture the panoramic room-object feature using an LSTM:

$$h_t^v = \text{LSTM}_v(g_t, h_{t-1}^v), \forall t = 1, \ldots, N \quad (6)$$

5.1 Trajectory Encoder

The trajectory encoder consists of a multilayer bidirectional LSTM to summarise the input sequence at each time step conditioned on the navigational trajectory. This bidirectional approach ensures action context at each step is influenced by both the historic and the future actions in the sequence.

The first BiLSTM_A encodes navigation actions from the ground truth action a. Scaled-dot attention is applied to the action hidden state h^a and the panoramic visual features f_v giving a context vector c_w. A second LSTM encodes the change of the context vector as $h_t^{v,a}$. This hidden state is used by the decoder to learn visual and language alignment. Formally:

$$h_t^a = \text{BiLSTM}_A(a_t, h_{t-1}^a) \quad (7)$$

$$\alpha_w = \text{softmax}\left(\frac{Q_a(h_t^a)K_v^T(h_t^v)}{\sqrt{D_k}}\right) \quad (8)$$

$$c_w = \sum_{i,j} \alpha_w h_t^a \quad (9)$$

$$\tilde{h} = \tanh(\mathbf{W}[c_w; h_t^v]) \quad (10)$$

$$\hat{h}_t^{v,a} = \text{BiLSTM}_{VA}(c_w, \tilde{h}) \quad (11)$$

where a is the action embedding, h is the hidden state. Also, $Q_a = F_q^a(h_t^a)$, $K_v = F_k^v(f_v)$, and D_k are query and key vectors, hidden dimension size of the soft attention, respectively.

5.2 Instruction Decoder

Our instruction decoder is guided by semantic and structural knowledge from the environment. The basic structure of the decoder is as follows. When the decoder is provided with the previous instruction token and the visual-action context $h_t^{v,a}$, it applies an LSTM to encode the instruction token embedding from the previous time step:

$$w_{t-1}^{emb} = \text{embedding}(w_{t-1}) \quad (12)$$

$$h_t^X = \text{LSTM}_X(w_{t-1}^{emb}, h_{t-1}^{v,a}) \quad (13)$$

we apply a scaled-dot attention on the projected instruction context $Q_X = F_q(h_t^X)$ and the vision-action context $K_{va} = h_t^{v,a}$ and $V = F_v(h_t^X)$:

$$\hat{h}_t^{vaX} = \text{Attention}(Q_X, K_{va}, V_X) \quad (14)$$

Finally, the next predicted word is the token of maximum probability:

$$w_t = \arg\max(\mathbf{W}\hat{h}_t^{vaX}) \quad (15)$$

6 Training

The Encoder (§5.1) and Decoder (§5.2) modules of SAS are trained end-to-end. SAS model predicts the next token w_t based on the complete trajectory $T = \{O_1, \ldots, O_N\}$ and all previous tokens $w_{<t}$. The trajectory embedding from the encoder and $w_{<t}$ is fed to the decoder to produce a probability distribution p_L over the next word token. This distribution is sampled as in (15) to predict the next token w_t.

SAS model is trained using a mixture of a Teacher-Forcing (TF) (Williams and Zipser, 1989) and the ARL method (§6.1). In TF, the decoder generates the next token based on a ground truth token instead of using its predicted token as in the Student Forcing (SF) strategy. This method has been shown to improve the baseline methods (Anderson et al., 2018). Next, we describe reward learning in detail.

6.1 Reward Learning

Applying Reinforcement Learning (RL) for an instruction generation task using automatic evaluation metrics as reward functions, causes the model to *game* the metrics. Instead, in reward learning,

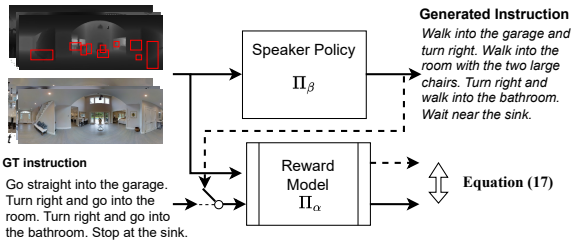


Figure 3: Adversarial training of SAS model. SAS learns to generate instruction, while reward model learns the reward function from ground truth data. The learned reward function is employed to optimise the policy

a reward model learns the best reward for human-annotated and speaker-generated instructions. In this strategy, we use a Generative Adversarial Network (GAN) (Goodfellow et al., 2020) architecture - with a Policy and Reward model - to learn an association between the instruction and reward distributions (Fig. 3). In an RL sense, this is a Markov decision process (MDP) with SAS as a policy $G_\beta(\cdot)$, generating words X^G and receiving a *reward* $r(X^G)$ score. The objective is to maximise the expected reward of G_β, $\mathbb{E}_{X^G \sim G_\beta}[r(X^G)]$. In reward learning, the reward distribution is learned from the demonstrations, rather than adopting a function to provide a reward.

We adapt the idea of Reward-Boltzmann distribution from Wang et al. (2018) to approximate a reward obtained for identifying *fake* (speaker-generated) or *real* (human-annotated) instructions. The approximate reward distribution Π_α for an instruction X is defined as:

$$\Pi_\alpha(X) = \frac{e^{R_\alpha(X)}}{\sum_X e^{R_\alpha(X)}} \quad (16)$$

where R_α is the reward function. The optimal reward function is achieved when (16) is equal to the distribution of human-annotated instructions. We optimise this using an adversarial two player min-max game between (1) Π_α maximising its similarity (measured by KL-divergence) with the empirical distribution $\Pi_\epsilon(X)$ of the training dataset and minimising its similarity with the distribution of *fake* instructions from the policy Π_β and (2) maximising the similarity of the policy distribution Π_β with that of Π_α. Formally, the objective is:

$$\max_\beta \min_\alpha KL(\Pi_\epsilon||\Pi_\alpha) - KL(\Pi_\beta||\Pi_\alpha) \quad (17)$$

This is optimised through a policy-gradient-based reinforcement learning method.

Reward Model We investigate two reward models R_ω based on the CNN and RNN models. The CNN-based discriminator uses the GloVe embedded instruction $\mathbf{X}_{emb}$ and visual feature Oi to produce a reward score R_ω after the activation function. Formally, the CNN and RNN rewards are, respectively:

$$R_\omega^{CNN} = \mathbf{W}_R(\text{Conv}(X_{emb}); \mathbf{W}_O O_i) \quad (18)$$
$$R_\omega^{RNN} = \mathbf{W}_R(X_{emb}; \mathbf{W}_O \text{AvgPool}(O_i)) \quad (19)$$

where W_R and W_O are linear learnable weights, $Conv$ represents the convolution layers followed by the mean pooling operation, and $[;]$ is the feature concatenation. The final sigmoid activation is not shown for brevity.

Both the Speaker policy and the reward models are trained alternately using the Adversarial Reward Learning (ARL) algorithm (Algorithm A). Reward models are evaluated in our ablation studies (§8.3).

6.2 Supervised Learning

The supervised learning objectives used for both the teacher-forcing (used in the final model) and the student-forcing (for the ablation study) strategies are as follows.

Language modelling Conditioned on path T and linguistic embedding $w_{<t}$, the probability of decoded words is optimised as a maximum likelihood estimation (MLE) problem:

$$\mathcal{L}_{LM} = -\sum_{t \in \{1:N\}} \log(p_L(w_t|w_{<t}, T)) \quad (20)$$

where p_L is the likelihood of a token given trajectory T and ground truth instruction tokens $w_{<t}^{GT}$.

Unlikelihood training Even low-perplexity machine generation models are prone to repeating tokens when presented with small examples (Holtzman et al., 2020). To mitigate this, we apply Sequence-Level unlikelihood loss (Welleck et al., 2020) on the decoded instruction that penalises repetition of word tokens w. The objective is to minimise the logarithmic likelihood of negative candidates (repeated tokens) $\mathcal{C}^t$ conditioned on previous tokens $w_{<t}$:

$$\mathcal{L}_{ULS} = -\sum_{c \in \mathcal{C}^t} \log(1 - p_\mu(c|w_{<t})) \quad (21)$$

Table 1: Benchmarking results of Speaker-based models (§7) on R2R dataset

Methods	R2R ValSeen					R2R ValUnseen				
	SPICE	CIDEr	METEOR	ROUGE	BLEU-4	SPICE	CIDEr	METEOR	ROUGE	BLEU-4
Speaker-Follower (Fried et al., 2018)	22.1	43.7	23.0	49.5	28.3	18.9	37.9	21.7	48.0	26.3
EnvDrop (Tan et al., 2019)	24.3	47.8	24.5	49.6	27.7	21.8	41.7	23.6	49.0	27.1
CCC (Wang et al., 2022)	23.1	**54.3**	23.6	49.3	28.7	21.4	**46.1**	23.1	47.7	27.2
LANA (Wang et al., 2023)	25.6	53.3	24.5	50.3	31.4	22.6	45.7	23.8	49.8	29.8
SAS_{TF} (Ours)	27.9	53.1	28.3	54.9	30.2	22.2	44.9	**26.3**	55.4	30.2
SAS_{ARL+TF} (Ours)	**28.1**	51.6	**29.7**	**56.8**	**31.4**	**24.8**	43.5	25.7	**56.5**	**33.8**

Table 2: Benchmarking results of Speaker-based models (§7) on R4R dataset

Methods	R4R ValSeen				R4R ValUnseen			
	SPICE	CIDEr	METEOR	ROUGE	SPICE	CIDEr	METEOR	ROUGE
Speaker-Follower (Fried et al., 2018)	16.4	9.9	21.3	45.3	20.7	13.9	17.2	35.9
EnvDrop (Tan et al., 2019)	20.9	21.6	24.5	47.3	21.8	20.0	18.7	36.3
CCC (Wang et al., 2022)	21.9	24.5	25.2	48.0	23.3	20.6	19.3	36.5
LANA (Wang et al., 2023)	24.5	**28.7**	26.1	48.4	26.2	**23.1**	20.0	37.6
SAS_{TF} (ours)	25.8	26.5	27.6	50.1	27.4	21.7	21.1	38.4
SAS_{ARL+TF} (ours)	**26.2**	22.3	**28.9**	**50.3**	**28.1**	22.5	**22.8**	**39.2**

Temporal Alignment Loss We introduce a temporal alignment loss (TAL) to train the decoder to attend between action phrases and visual-action context. The decoder's attention matrix A_D, which represents attention between word tokens and the panoramic action context, is compared to the ground truth vision language alignment scores (§4.2). Formally,

$$A_{GT} = \begin{cases} 1, & w \leftrightarrow o_i; w \in X, o_i \in O \\ 0, & \text{otherwise} \end{cases} \quad (22)$$

where $\leftrightarrow$ denotes action phrase-viewpoint alignment. $\mathcal{L}_{TAL}$ is the binary cross-entropy loss between A_{GT} and A_D.

Total Objective The total training objective is the weighted sum of all losses, that is, $\mathcal{L} = \lambda_{LM}\mathcal{L}_{LM} + \lambda_{ULS}\mathcal{L}_{ULS} + \lambda_{TAL}\mathcal{L}_{TAL}$.

7 Experiments

7.1 Datasets

Our method is assessed using two datasets from the Vision-and-Language Navigation (VLN) field: R2R, which features brief trajectory paths and instructions for locating rooms, and R4R, an extension of R2R that links two adjacent tail-to-head trajectories along with their associated instructions to produce longer instructions. For training, we augment R2R dataset with the Path Mixing (PM) dataset §4.1.

7.2 Evaluation Metrics

We evaluate the performance of SAS instruction generation using standard language metrics such as SPICE (Anderson et al., 2016), CIDEr (Vedantam et al., 2015), ROUGE (Lin, 2004), METEOR (Denkowski and Lavie, 2014) and BLEU-4 (Papineni et al., 2001). SPICE is considered the main metric in navigational instruction generation tasks (Zhao et al., 2021; Wang et al., 2022). A high SPICE score indicates high lexical and semantic similarities of sentences and higher success of an embodied agent, which is important for navigational instructions (Zhao et al., 2021).

7.3 Implementation Details

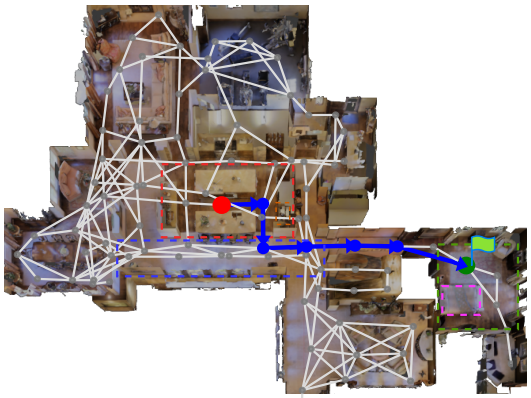
We use Speaker-Follower (Fried et al., 2018), a popular baseline used in previous work, for our experiments. The model is trained for 100k epochs (≈14h) using NVIDIA RTX A6000 with batch size 8 and AdamW (Loshchilov and Hutter, 2019) optimiser with learning rate 5e-4. The visual feature and GloVe embedding sizes are 2048 and 300, respectively. The hidden size for the attention layers D_k is 512. The training objective weights are set in the ratio $\lambda_{LM} : \lambda_{ULS} : \lambda_{TAL} = 2 : 1 : 1$ to prioritize language learning over repetition and alignment.. The hidden dimensions for the two-layer BiLSTM and the QKV sizes in Attention are set to 768d. We report the evaluation values of a single run.

8 Results

We evaluate two variations of SAS to measure the effectiveness of the proposed method (1) SAS with teacher-forced SAS_{TF} using PM augmentation and temporal alignment (TAL) and (2) SAS_{ARL+TF} a mixture of TF and ARL using all augmentation and supervised learning objectives. From the results (Table 1), we see that our SAS method improves on the baseline by a large margin. Reward learning has absolute improvements of +5.9 (SPICE), +5.6 (CIDEr), +4 (METEOR), +8.5 (ROUGE), and +7.5 (BLEU-4) in the R2R ValUnseen split compared to the Speaker-Follower baseline. In the long instruction dataset R4R (Table 2), both variations show better scores on all metrics (SPICE: +7.4, METEOR: +8.6, ROGUE: +3.3 and CIDEr +8.6). In both `ValUnseen` splits, the CIDEr score is markedly impacted (R2R: -2.6 and R4R: -0.6) when comparing the overall best model to the baseline models.

8.1 Discussion

SAS shows a better instruction generation capability with TAL, which goes to show that supervising the speaker with action-only sentences is useful. In addition, our method outperforms the baseline in most of the major metrics, specifically in SPICE. This shows that spatial awareness is beneficial for Speaker models. In the R2R and R4R datasets, both SAS_{TF} and SAS_{ARL+TF} have the best scores compared to the previous models, except for CIDEr. CIDEr rewards lexical similarity over semantic similarity. As temporal alignment of actions is pertinent in instruction generation and not in the lexical order, lower scores in these metrics definitely do not reflect wrong actions. Among these metrics, a higher SPICE score shows that our model generates temporally consistent instructions. Zhao et al. (2021) observe that SPICE correlates with human way-finding performance, VLN agent navigation performance, and subjective human judgements of instruction quality, when averaged over many instructions. This correlation is not observed at the instruction level due to the high variance between the words used in the instructions. Human evaluation of generated instructions should be performed to ensure the actual quality of the instruction. In our study, we consider the high SPICE score as an early indicator of robust pathfinding performance for both agents and humans. It also reflects that human judgement of the quality of SAS-generated instructions is also high.



GT: *Walk through the kitchen passed the sink and around the corner out into the hallway. Walk into the arched entry to the left of the stairwell. Continue into the room with the armchair and bed.*
Speaker-Follower: *Exit the kitchen. Walk to the hallway and turn left. Walk into the room.*
SAS: *Walk through the kitchen past the oven and into the hallway. Walk through the hallway on the left into the bedroom. Wait near the bed.*

Figure 4: An example of a trajectory and the corresponding generated instruction using SAS_{ARL+TF} model.

8.2 Qualitative Results

Our SAS model is able to generate meaningful instructions by including object and scene relevant tokens (such as "oven") as shown in Fig. 4 that are not referenced in the ground-truth instruction, while the Speaker-Follower baseline model produces shorter and action-focused sentences.

8.3 Ablation Studies

Table 3: Ablation Study (§8.3) on R2R dataset `ValUnseen` split

Met.	PM	TAL	CNN	GRU	SPICE	CIDEr	METEOR	ROUGE	BLEU-4
#1					22.5	38.0	23.9	48.3	26.2
#2					22.8	39.1	23.9	49.5	27.3
#3	✓				21.2	39.9	23.6	50.1	27.8
#4		✓			22.1	40.4	24.1	53.2	28.6
#5	✓	✓			22.2	44.9	26.3	55.4	30.2
#6	✓	✓	✓		24.3	42.9	26.3	54.4	30.3
#7	✓	✓		✓	24.8	43.5	25.7	56.5	33.8

Here we ablate on different augmentation and training methods (Table 3) evaluated on R2R `ValUnseen` split. Method #1 uses the aforementioned Student forcing (SF) and #2 represents Teacher forcing (TF) for training. When Path Mixing is applied to the SAS model (#3), the Speaker learns the frequent object tokens in the instruction and how to correlate them with the visual features. Models #4 and #5 (SAS_{TF}), trained using TAL, learn to associate the object tokens with the actions from the trajectory and inversely co-relate navigational actions with action phrases in the ground

truth instructions (*Walk through the double door...*). The models #6 and #7 (SAS_{ARL+TF}) are trained to improve the SPICE score using ARL. Using ARL and a GRU-based reward model (#7) has an advantage over using the CNN-based reward model (#6), which produces the highest scores.

8.4 Spatial and Semantic effectiveness

To study the effectiveness of the proposed spatial and semantic encoding, we measure the amount of object and spatial phrases mentioned in the generated instructions for the speakers evaluated in the R2R `ValUnseen` environment.

Table 4: Spatial and Semantic referrals (§8.4) on R2R `ValUnseen` environment

Method	Obj.	Act.	NonStop
Human	14123 (6.01)	8933 (3.80)	36689 (15.61)
LANA	2861 (3.62)	2842 (3.62)	9685 (12.36)
SAS	3379 (4.31)	3184 (4.06)	10790 (13.78)

In Table 4, the values outside the parentheses represent the total counts of objects/landmarks (Obj.) i.e. *chair, bathroom, etc.*, actions/direction phrases (Act.) i.e. *turn left, top of, etc.*, and non-stopwords (NonStop). Meanwhile, the values in parentheses denote the average number of entities per instruction for each respective category. We observed that the SAS speaker includes 18.11% more objects and landmark entities and 12.03% more action/direction phrases compared to the LANA speaker. Furthermore, the average length of the instructions is also higher for SAS, indicating richer or more detailed instructions. Although the SAS model did not refer to all the objects or landmarks in the ground truth instructions (SAS: 4.31, Human: 6.01), it includes more action/direction phrases (SAS: 4.06, Human: 3.80). This suggests a better specificity for actions and spatial awareness.

8.5 Limitations

Large-scale datasets featuring a broad variety of human-annotated navigation instructions are rare, presenting a significant challenge in the field. Our approach seeks to navigate this obstacle by leveraging a small-scale dataset originally compiled with a different objective: to facilitate the learning of navigation from instructions. It is important to consider this context when evaluating our method's performance, as it operates under the constraint of limited data diversity and volume. Next, we explain some of the challenges in the dataset.

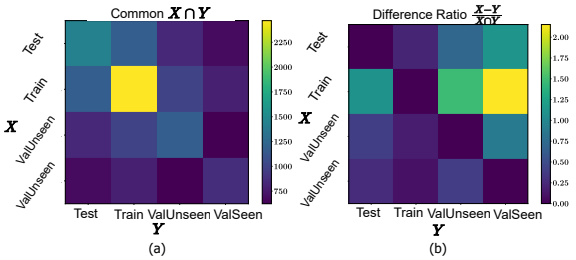


Figure 5: Unique instruction words present in R2R dataset splits. (a) common words between splits, (b) shows the ratio of number of different words to number of common words in between the splits.

R2R dataset exhibits a notable variation in unique tokens across its different splits: `Train`, `ValSeen`, `ValUnseen`, and `TestUnseen`. Figure 5 (a) underscores the differences in token commonality across these splits, the disparities being largely attributed to the frequency of tokens in each split. Figure 5 (b) shows the ratio of different words to that of common words in each split, revealing that the `Train` split, in particular, contains a significant number of unique word tokens compared to common words with respect to other splits. This variation poses a unique challenge for our SAS model, which aims to mimic the instruction distribution of the training set, but may diverge from the linguistic characteristics of the `ValSeen` and `ValUnseen` splits, potentially negatively impacting evaluation scores.

9 Conclusion

This work proposes a novel navigation instruction generation model that can produce diverse instructions by attending to several structural and semantic cues from the environment. By providing objects, their locations, and their relationships from the scene, Spatially-Aware Speaker can refer to important aspects of the scene in the instruction. An adversarial reward learning method encourages the model to generate diverse instructions. The results show that our method improves on the standard evaluation metrics and performs better than the baselines.

Future work In future work, integrating the SAS model with multimodal transformer architectures will be crucial for enhancing the generation of open-vocabulary embodied instructions. This direction promises to overcome the limitations posed by current dataset constraints and improve performance in instruction generation tasks.

References

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. SPICE: Semantic propositional image caption evaluation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 9909 LNCS, pages 382–398.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sunderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3674–3683. IEEE.

Kushal Arora, Layla El Asri, Hareesh Bahuleyan, and Jackie Cheung. 2022. Why exposure bias matters: An imitation learning perspective of error accumulation in language generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 700–710, Dublin, Ireland. Association for Computational Linguistics.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niebner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3d: Learning from rgb-d data in indoor environments. In *2017 International Conference on 3D Vision (3DV)*, pages 667–676.

Andrea F Daniele, Mohit Bansal, and Matthew R Walter. 2017. Navigational instruction generation as inverse reinforcement learning with neural machine translation. In *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, pages 109–118.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Vishnu Sashank Dorbala, James F. Mullen Jr, and Dinesh Manocha. 2023. Can an Embodied Agent Find Your "Cat-shaped Mug"? LLM-Based Zero-Shot Object Navigation . *IEEE Robotics and Automation Letters*, pages 1–8.

Zi Yi Dou and Nanyun Peng. 2022. FOAM: A Follower-aware Speaker Model For Vision-and-Language Navigation. *NAACL 2022 - 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference*, pages 4332–4340.

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*, volume 2018-Decem, pages 3314–3325.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative adversarial networks. *Commun. ACM*, 63(11):139–144.

Muraleekrishna Gopinathan, Jumana Abu-Khalaf, David Suter, Sidike Paheding, and Nathir A. Rawashdeh. 2023. What Is Near?: Room Locality Learning for Enhanced Robot Vision-Language-Navigation in Indoor Living Environments. In *ArXiv*.

Muraleekrishna Gopinathan, Giang Truong, and Jumana Abu-Khalaf. 2021. Indoor semantic scene understanding using 2d-3d fusion. In *2021 Digital Image Computing: Techniques and Applications, DICTA 2021, Gold Coast, Australia, November 29 - December 1, 2021*, pages 1–8. IEEE.

W. Andrew Harrell, Jeffrey W. Bowlby, and Deana Hallhoffarth. 2000. Directing wayfinders with maps: The effects of gender, age, route complexity, and familiarity with the environment. *The Journal of Social Psychology*, 140(2):169–178.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The Curious Case of Neural Text DeGeneration. In *8th International Conference on Learning Representations, ICLR 2020*.

Yicong Hong, Cristian Rodriguez, Qi Wu, and Stephen Gould. 2020. Sub-Instruction Aware Vision-and-Language Navigation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3360–3376, Stroudsburg, PA, USA. Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spacy: Industrial-strength natural language processing in python.

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. 2020. Stay on the path: Instruction fidelity in vision-and-language navigation. *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*, pages 1862–1872.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. 2017. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *International Journal of Computer Vision*, 123(1):32–73.

Benjamin Kuipers. 1978. Modeling spatial knowledge. *Cognitive Science*, 2(2):129–153.

Chia Wen Kuo, Chih Yao Ma, Judy Hoffman, and Zsolt Kira. 2023. Structure-Encoding Auxiliary Tasks for Improved Visual Representation in Vision-and-Language Navigation. In *Proceedings - 2023 IEEE Winter Conference on Applications of Computer Vision, WACV 2023*, pages 1104–1113.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chong Liu, Fengda Zhu, Xiaojun Chang, Xiaodan Liang, Zongyuan Ge, and Yi-Dong Shen. 2021. Vision-Language Navigation with Random Environmental Mixup. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1624–1634.

Gary Look, Buddhika Kottahachchi, Robert Laddaga, and Howard Shrobe. 2005. A location representation for generating descriptive walking directions. In *International Conference on Intelligent User Interfaces, Proceedings IUI*, pages 122–129.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*.

Abhinav Moudgil, Arjun Majumdar, Harsh Agrawal, Stefan Lee, and Dhruv Batra. 2021. SOAT: A Scene and Object-Aware Transformer for Vision-and-Language Navigation. *Advances in Neural Information Processing Systems*, 9(NeurIPS):7357–7367.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02*, volume 371, pages 311–318, Morristown, NJ, USA. Association for Computational Linguistics.

Yuankai Qi, Zizheng Pan, Shengping Zhang, Anton van den Hengel, and Qi Wu. 2020. Object-and-Action Aware Model for Visual Language Navigation. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 12355 LNCS, pages 303–317.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2017. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 39(06):1137–1149.

Chan Hee Song, Jihyung Kil, Tai-Yu Pan, Brian M. Sadler, Wei-Lun Chao, and Yu Su. 2022. One Step at a Time: Long-Horizon Vision-and-Language Navigation with Milestones. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15461–15470. IEEE.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI Conference on Artificial Intelligence*, pages 4444–4451.

Hao Tan, Licheng Yu, and Mohit Bansal. 2019. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, volume 1, pages 2610–2621, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4566–4575.

Hanqing Wang, Wei Liang, Jianbing Shen, Luc Van Gool, and Wenguan Wang. 2022. Counterfactual cycle-consistent learning for instruction following and generation in vision-language navigation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15450–15460, Los Alamitos, CA, USA. IEEE Computer Society.

Xiaohan Wang, Wenguan Wang, Jiayi Shao, and Yi Yang. 2023. LANA: A Language-Capable Navigator for Instruction Following and Generation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2023-June, pages 19048–19058.

Xin Wang, Wenhu Chen, Yuan Fang Wang, and William Yang Wang. 2018. No metrics are perfect: Adversarial reward learning for visual storytelling. In *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, volume 1, pages 899–909.

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 *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 6622–6631. IEEE.

Shawn L. Ward, Nora Newcombe, and Willis F. Overton. 1986. Turn left at the church, or three miles north: A study of direction giving and sex differences. *Environment and Behavior*, 18(2):192–213.

Sean Welleck, Ilia Kulikov, Stephen Roller, Emily Dinan, Kyunghyun Cho, and Jason Weston. 2020. Neural Text Degeneration With Unlikelihood Training. *8th International Conference on Learning Representations, ICLR 2020*, i:1–17.

Ronald J. Williams and David Zipser. 1989. A learning algorithm for continually running fully recurrent neural networks. *Neural Computation*, 1:270–280.

Ming Zhao, Peter Anderson, Vihan Jain, Su Wang, Alexander Ku, Jason Baldridge, and Eugene Ie. 2021. On the evaluation of vision-and-language navigation instructions. In *Proceedings of the 16th Conference*

of the European Chapter of the Association for Computational Linguistics: Main Volume, pages 1302–1316, Online. Association for Computational Linguistics.

Supplementary material for the manuscript titled *"Spatially-Aware Speaker for Vision Language Navigation Instruction Generation"*

Appendix A Adversarial Reward Learning algorithm

We use Reward Learning (ARL) extended from (Wang et al., 2018). We alternatively train SAS policy and reward models for 100 epochs each. The speaker policy is optimised using the loss functions (§6) explained in the main text.

Algorithm 1 Adversarial Reward Learning Algorithm

for epoch ← 1 to K **do**
  Obtain instruction $X \leftarrow \Pi_\omega$ ▷ SAS Speaker Policy
  if Train-Policy **then**
    Obtain instruction $\tilde{X} \leftarrow D$
    Update Speaker Policy gradient
  else if Train-Reward **then**
    Update Reward gradient ▷ CNN or GRU
  end if
end for

Appendix B Algorithm to extract action phrases from instruction

Algorithm 2 Algorithm to extract action phrases

Input: Navigation Instruction X
Output: Action phrases $l_{\boldsymbol{X}}$
Initialise the empty action phrase list $l_{\boldsymbol{X}}$.
$l_{typ} \leftarrow$ {TRANVERB, DITRANVERB, INTRANVERB, NOUN}
$l_{pos} \leftarrow$ {ADV, ADP, INTJ, DET, INTRANVERB}
for w_j, w_{j+1} in X **do**
  $c_{w_j} \leftarrow$ **check_verb**(w_j)
  $c_{w_{j+1}} \leftarrow$ **check_verb**(w_{j+1})
  if c_{w_j} in l_{typ} **then**
    if $c_{w_{j+1}}$ in l_{pos} **then**
      $l_{\boldsymbol{X}} \leftarrow (w_j; w_{j+1})$ ▷ Join words as a phrase
    else if $c_{w_j} \neq$ NOUN **then**
      $l_{\boldsymbol{X}} \leftarrow w_j$ ▷ Verb as a phrase
    end if
  end if
end for

Appendix C Implementation Details

We implement our method with PyTorch library. The instruction speaker is trained with AdamW Optimizer (Loshchilov and Hutter, 2019) for 100k iterations. Consistent with previous work, panoramic visual features are extracted using a ResNet-152 model, and the angle feature used is 128 dimensional. We retrieve $K = 6$ objects and relationships from Visual Genome for each view direction. The dimension of the hidden state is set to 512. All experiments were performed on an NVIDIA RTX A6000 GPU. SAS_{TF} model is trained to 100K iterations. The SAS_{TF+ARL} model uses pre-trained SAS_{TF} weights for initialisation. In ARL strategy, policy and reward models are trained alternatively every 100 iterations. All parameters were chosen based on the performance on the `ValUnseen` split. SAS model has 14.3M learnable parameters. The losses are mixed using the weights $\lambda_{LM} : \lambda_{ULS} : \lambda_{TAL}$ is $2 : 1 : 1$. The implementation of the metrics is obtained from the publicly available COCO Caption Evaluation Toolkit [3]. spaCy package uses the `en_core_web_lg` model.

C.1 Hyperparameter Search

The hyper parameters for the final model are selected from the highest SPICE score on the `ValUnseen` split. The learning rate is fixed based on observing the reward, IRL scores, and supervision losses to prevent model from over-fitting.

C.2 Dataset Statistics

R2R It has four splits, i.e., `Train` (61 scenes, 14k instructions), `ValSeen` (61 scenes, 1k instructions), `ValUnseen` (11 scenes, 2k instructions), and `TestUnseen` (18 scenes, 4k instructions). `Train` and `ValSeen` splits share house scenes but not with the unseen splits.

R4R It extends R2R and contains three sets, i.e., `Train` (61 scenes, 233k instructions), `ValSeen` (61 scenes, 1k instructions), and `ValUnseen` (11 scenes, 45k instructions).

Appendix D Qualitative Examples

Here we list some successful instructions generated by the SAS model against the speaker follower (Fried et al., 2018) baseline.

[3] https://github.com/tylin/coco-caption

D.1 Successful Example 1

Baseline: *Walk into the garage and turn right. Turn right and walk into the room. Walk into the bathroom. Stop.*
SAS$_{TF}$: *Walk into the garage and turn right. Walk into the room with the two large chairs. Turn right and walk into the bathroom. Wait near the sink.*
SAS$_{ARL+TF}$: *Walk through the garage and turn right at the cupboard. Walk into the living room towards the couches. Walk into the bathroom and wait near the sink.*
GT: *Go straight into the garage. Turn right and go into the room. Turn right and go into the bathroom. Wait near the sink.*

D.2 Successful Example 2

Baseline: *Walk the stairs.*
SAS$_{TF}$: *Walk out of the bedroom. turn left and walk up the stairs . stop on the second step from the bottom .*
SAS$_{ARL+TF}$: *Walk past the television and out of the bedroom. Turn left and walk up the stairs. Stop on the third step from the bottom.*
GT: *Walk through the bedroom and out into the hall way. Turn left and walk up to the stairs. Walk up to the first step and stop.*

Appendix E More visualisation of trajectories

E.1 Successful Example 1

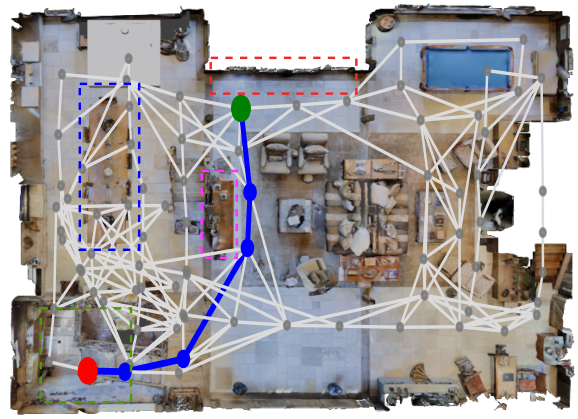
Figure 6 shows one of the success cases. Here, SAS_{TF} generated the correct transition phrases, but failed to mention any stop locations. On the other hand, SAS_{ARL+TF} refers to objects outside of reference sentences and also identifies the correct stop location. This is a good example of Panoramic Room-Object Attention in working.

E.2 Failure Example 1

Figure 7 shows one of the failure cases. Here both SAS models fail to generate the correct instruction even though the rooms (living room, kitchen, dining room) and objects (table, chair) are identified. The models failed to align the objects, rooms and actions in the correct sequence.

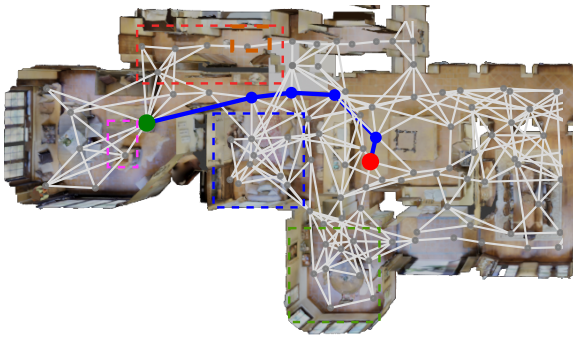
E.3 Learning Curves

Figure 8 shows the average ARL rewards and SPICE scores from training the model with CNN and GRU reward models. Both models approach a reward close to 1, while the GRU reward model



GT: *Continue up the stairs, walk towards the sitting area, go straight passed the table on the left. stop by the double doors.*
SAS_{TF}: *Walk past the fireplace and up the stair. Walk past the dining table and through the doorway. Walk past the dining table and through the doorway.*
SAS_{ARL+TF}: *Walk up the stairs and turn left. Walk past the dining room table and chairs . Stop in front of the door .*

Figure 6: Example of a Successful Instruction Generation.



GT: *Walk towards the ovens and take a left. Walk towards the fireplace and enter the dining room to the right of the fireplace. Stop in front of the white chair.*
SAS_{TF}: *walk past the kitchen and into the kitchen . walk past the dining room table and chair . stop in front of the couch .*
SAS_{ARL+TF}: *Walk through the kitchen and turn left. Walk through the dining room and into the living room. Stop in the doorway to the living room .*

Figure 7: Example of a Failed Instruction Generation.

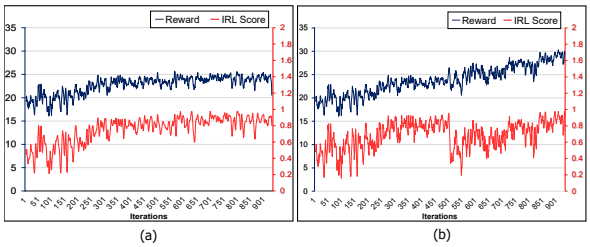


Figure 8: SPICE Scores and Rewards for (a) CNN-based and (b) GRU-based reward models.

obtains a higher SPICE score. This shows that recurrence can help the discriminator learn the difference between policy-generated and ground-truth instructions. A CNN can extract global information from the instruction but loses temporal information. As temporal information is crucial for navigational instructions and GRU can encode temporal aspects of the instruction as well as visual observations, it produces a higher reward for instructions closer to the ground truth.

Appendix F Ethical Considerations

Embodied navigation stands as a promising frontier with the potential to revolutionise the landscape of language understanding for robots, thus facilitating their seamless integration into everyday human life. However, any effort that involves human-robotic interaction requires a steadfast commitment to upholding ethical, privacy, safety, and legal standards. Although the metrics employed to assess our work align with those commonly used in the machine generation domain, further investigations are imperative to ensure the ethical and safety considerations associated with the instructions generated using automatic methods and their use in the real world.

Our research draws on the R2R (Anderson et al., 2018), FGR2R (Hong et al., 2020) and R4R (Jain et al., 2020) datasets under the MIT licence, which feature an extensive collection of indoor photos captured from American houses, licenced by Matterport3D[4]. The Matterport3DSimulator, used in our experiments, is also under MIT license. To protect privacy and confidentiality, the providers of the datasets have anonymised both the houses and the associated photos. In addition, the navigational instructions derived from these datasets are devoid of explicit language. As a result, our work shows minimal ethical, privacy, or safety concerns.

[4] https://kaldir.vc.in.tum.de/matterport/MP_TOS.pdf