Counterfactual Vision-and-Language Navigation via Adversarial Path Sampler

Tsu-Jui Fu[†], Xin Eric Wang[‡], Matthew F. Peterson[†],
Scott T. Grafton[†], Miguel P. Eckstein[†], William Yang Wang[†]

[†]UC Santa Barbara
tsu-juifu@ucsb.edu, {peterson, scott.grafton,
miguel.eckstein}@psych.ucsb.edu, william@cs.ucsb.edu
[‡]UC Santa Cruz
xwang366@ucsc.edu

Abstract. Vision-and-Language Navigation (VLN) is a task where agents must decide how to move through a 3D environment to reach a goal by grounding natural language instructions to the visual surroundings. One of the problems of the VLN task is data scarcity since it is difficult to collect enough navigation paths with human-annotated instructions for interactive environments. In this paper, we explore the use of counterfactual thinking as a human-inspired data augmentation method that results in robust models. Counterfactual thinking is a concept that describes the human propensity to create possible alternatives to life events that have already occurred. We propose an adversarial-driven counterfactual reasoning model that can consider effective conditions instead of low-quality augmented data. In particular, we present a model-agnostic adversarial path sampler (APS) that learns to sample challenging paths that force the navigator to improve based on the navigation performance. APS also serves to do pre-exploration of unseen environments to strengthen the model's ability to generalize. We evaluate the influence of APS on the performance of different VLN baseline models using the room-to-room dataset (R2R). The results show that the adversarial training process with our proposed APS benefits VLN models under both seen and unseen environments. And the pre-exploration process can further gain additional improvements under unseen environments.

1 Introduction

Vision-and-language navigation (VLN) [3,8] is a complex task that requires an agent to understand natural language, encode visual information from the surrounding environment, and associate critical visual features of the scene and appropriate actions with the instructions to achieve a specified goal (usually to move through a 3D environment to a target destination).

To accomplish the VLN task, the agent learns to align linguistic semantics and visual understanding and also make sense of dynamic changes in vision-and-language interactions. One of the primary challenges of the VLN task for artificial agents is data scarcity; for instance, while there are more than 200K

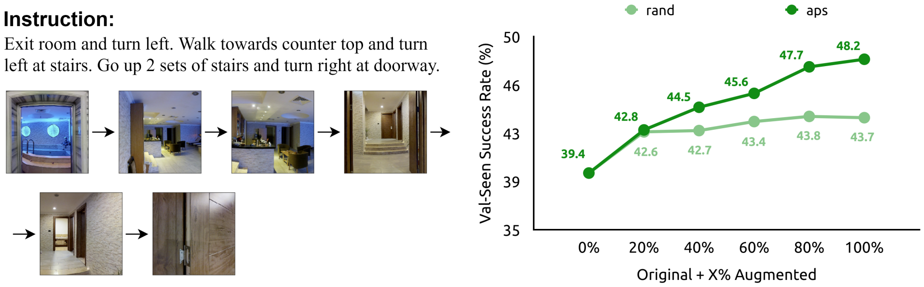


Fig. 1. The comparison between randomly-sampled (rand) and APS-sampled (aps) under validation-seen set for Seq2Seq over different ratios of augmented path used.

possible paths in the Room-to-Room (R2R) dataset [3], the R2R training data comprises only 14K sampled paths. This scarcity of data makes learning the optimal match between vision and language within the interactive environments quite challenging.

Meanwhile, humans often lack extensive experience with joint access to visual experience and accompanying language instructions for navigating novel or unfamiliar environments, yet the human mind can navigate environments despite this data scarcity by incorporating mechanisms such as counterfactual reasoning [29] and self-recovered missing information. For example, if a human follows an instruction to “turn right” and they see a door in front of them, they can also consider what they may have encountered had they turned left instead. Or, if we stop in front of the dining table instead of walking away from it, what should the instruction be? The premise, then, is that counterfactual reasoning can improve performance in a VLN task through exploration and consideration of alternative actions that the agent did not actually make. This may allow the agent to operate in data-scarce scenarios by bootstrapping familiarity of environments and the links between instructions and multiple action policy options.

Counterfactual thinking has been used to increase the robustness of models for various tasks [24,12]. However, no explicitly counterfactual models have been applied to the VLN task specifically. Speaker-Follower [11], which applies a back-translated speaker model to reconstruct the instructions for randomly-sampled paths as augmented training examples, is probably the VLN model that comes closest to instantiating counterfactual thinking.

While the use of augmented training examples by the Speaker-Follower agent resembles a counterfactual process, the random sampling method is too arbitrary. Fig. 1 reports the performance of the model trained with randomly-sampled augmented data (the line in a light color) over different ratios of the augmented path used. It shows that the success rate stops increasing once augmented paths account for 60% or more of the training data [19]. Since those paths are all randomly sampled, it can limit the benefit of counterfactual thinking to data augmentation.

In this paper, we propose the use of adversarial-driven counterfactual thinking where the model learns to consider effective counterfactual conditions instead of sampling ample but uninformative data. We introduce a model-agnostic adversarial path sampler (APS) that learns how to generate augmented paths for training examples that are challenging, and thus effective, for the target navigation model. During the adversarial training process, the navigator is trying to accomplish augmented paths from APS and thus optimized for a better navigation policy, while the APS aims at producing increasingly challenging paths, which are therefore more effective than randomly-sampled paths.

Moreover, empowered by APS, the model can adapt to unseen environments in a practical setting—environment-based pre-exploration, where when deployed to a new environment, the robot can first pre-explore and get familiar with it, and then perform natural language guided tasks within this environment.

Experimental results on the R2R dataset show that the proposed APS can be integrated into a diverse collection of VLN models, improving their performance under both seen and unseen environments. In summary, our contributions are four-fold:

- We integrate counterfactual thinking into the vision-and-language navigation task, and propose the adversarial path sampler (APS) to progressively sample challenging and effective paths to improve the navigation policy.
- The proposed APS method is model-agnostic and can be easily integrated into various navigation models.
- Extensive experiments on the R2R dataset validate that the augmented paths generated by APS are not only useful in seen environments but also capable of generalizing the navigation policy better in unseen environments.
- We demonstrate that APS can also be used to adapt the navigation policy to unseen environments under environment-based pre-exploration.

2 Related Work

Vision-and-Language Navigation Navigation in 3D environments based on natural language instruction has recently been investigated by many studies [3,8,21,22,26,20,33,25,11,32,31,16]. For vision-and-language navigation (VLN), fine-grained human-written instructions are provided as guidance to navigate a robot in indoor environments. But data scarcity is a critical issue in VLN due to the high cost of data collection.

In order to augment more data for training, the Speaker-Follower model [11] applies a back-translated speaker model to generate instructions for randomly-sampled paths. In spite of obtaining some improvements from those extra paths, a recent study [19] shows that only a limited number of those augmented paths are useful and after using 60% of the augmented data, the improvement diminishes with additional augmented data. In this paper, we present a model-agnostic adversarial path sampler that progressively produces more challenging paths via an adversarial learning process with the navigator, therefore forcing the navigation policy to be improved as the augmented data grows.

Counterfactual Thinking Counterfactual thinking is a concept that describes the human propensity to create possible alternatives to life events that have already occurred. Humans routinely ask questions such as: “What if ...?” or “If there is only ...” to consider the outcomes of different scenarios and apply inferential reasoning to the process. In the field of data science, counterfactual thinking has been used to make trained models explainable and more robust [24,12,14]. Furthermore, counterfactual thinking is also applied to augment training targets [37,9,6]. Although previous studies have shown some improvements over different tasks, they all implement counterfactual thinking arbitrarily without a selection process to sample counterfactual data that might optimize learning. This can limit the effectiveness of counterfactual thinking. In this paper, we combine the adversarial training with counterfactual conditions to guide models that might lead to robust learning. In this way, we can maximize the benefit of counterfactual thinking.

Adversarial Training Adversarial training refers to the process by which two models try to detrimentally influence each other's performance and as a result, both models improve by competing against each other. Adversarial training has been successfully used to guide the target during model training [13,35,10,27,18,1]. Apart from leading the training target, adversarial training is also applied to data augmentation [5,36]. While previous studies just generate large amounts of augmented examples using a fixed pre-trained generator. In this paper, the generator is updated along with the target model and serves as a path sampler which samples challenging paths for effective data augmentation.

Pre-Exploration under Unseen Environments Pre-exploration under unseen environments is a popular method to bridge the gap between seen and unseen environments. Speaker-Follower [11] adopts a state-factored beam search for several candidate paths and then selects the best one. RCM [32] introduces self-imitation learning (SIL) that actively optimized the navigation model to maximize the cross-matching score between the generated path and the original instruction. Nevertheless, beam search requires multiple runs for each inference, and SIL utilizes the original instructions in the unseen environments for optimization. EnvDrop [31] conducts pre-exploration by sampling shortest paths from unseen environments and augments them with back-translation, which however utilizes the meta-information of unseen environments (*e.g.*, the shortest path planner that the robot is not supposed to use).

3 Methodology

3.1 Background

Visual-and-Language Navigation (VLN) At each time step t, the environment presents the image scene s_t. After stepping an action a_t, the environment will transfer to next image scene s_{t+1}:

$$s_{t+1} = \text{Environment}(s_t, a_t). \tag{1}$$

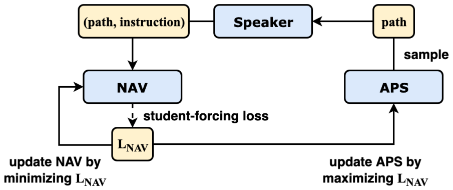


Fig. 2. The learning framework of our adversarial path sampler (APS), where Speaker denotes the back-translated speaker model and NAV denotes the navigation model.

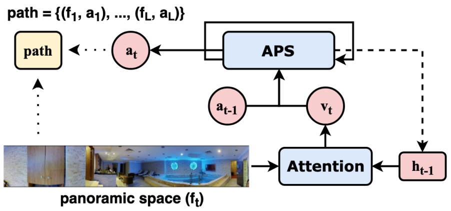


Fig. 3. The architecture of the adversarial path sampler (APS).

To carry out a VLN task, the navigation model steps a serious of actions $\{a_t\}_{t=1}^{T}$ to achieve the final goal described in the instruction. Though previous studies propose different architectures of navigation model (NAV), in general, NAV is a recurrent action selector based on the visual feature of the image scene, navigation instruction, and previous history:

$$\begin{aligned} f_t &= \text{VisualFeature}(s_t), \\ a_t &= \text{softmax}(\text{NAV}(f_t, I, h_t)), \end{aligned} \tag{2}$$

where f_t is the visual feature of the image scene s_t at time step t, I is the navigation instruction, h_t represents the previous history of image scenes, and a_t is the probability of each action to step at time step t. With a_t, we can decide which action to step based on greedy decoding (step the action with the highest probability).

In this work, we experiment under navigator with 3 different architectures, Seq2Seq [3], Speaker-Follower [11], and RCM [32].

Back-Translated Speaker Model Introduced in Speaker-Follower [11], the back-translated speaker model (Speaker) generates the instruction of a navigation path:

$$I = \text{Speaker}(\{(f_1, a_1), (f_2, a_2), ..., (f_L, a_L)\}), \tag{3}$$

where f_t is the visual feature of the image scene, a_t is the action taken at time step t, L represents the length of the navigation path, and I is the generated instruction. Speaker is trained with pairs of navigation paths and human-annotated instructions in the training data. With Speaker, we can sample various paths in the environments and augment their instructions.

3.2 Overview

The overall learning framework of our model-agnostic adversarial path sampler (APS) is illustrated in Fig. 2. At first, APS samples batch of paths P and we adopt the Speaker [11] to obtain the reconstructed instructions I. With the pairs of (P, I), we obtain the navigation loss $\mathcal{L}_{\text{NAV}}$. NAV minimizes $\mathcal{L}_{\text{NAV}}$ to improve navigation performance. While APS learns to sample paths that NAV can not perform so well by maximizing $\mathcal{L}_{\text{NAV}}$. Hence, there is an adversarial situation for $\mathcal{L}_{\text{NAV}}$ between NAV and APS, where APS aims at sampling challenging paths and NAV tries to solve the navigation tasks from APS.

By the above adversarial training process, we collect all of the (P, I) sampled from APS to compose the adversarial augmented data which can be more helpful to NAV than randomly-sampled one. Both Speaker and NAV are pre-trained using the original training set and Speaker keeps fixed during adversarial training. We collect all (P, I) sampled from APS as APS-sampled augmented path and further train NAV. More detail can be seen in Sec. 3.4.

3.3 Architecture of APS

As shown in Fig. 3, the proposed APS is a recurrent action sampler π_{APS} which samples series of actions $\{a_t\}_{t=1}^{T}$ (with the scene images $\{f_t\}_{t=1}^{T}$ presented from the environment) and combines as the path output, where f_t means the visual feature (e.g., extracted from the convolutional neural networks). For the panoramic image scene, $f_{t,j}$ represents the visual feature of the image patch at viewpoint j at time step t.

At each time step t, the history of previous visual feature and a_{t-1} is encoded as h_t by a long short-term memory (LSTM) [17] encoder:

$$h_t = \text{LSTM}([v_t, a_{t-1}], h_{t-1}), \tag{4}$$

where a_{t-1} is the action taken at previous step and v_t is the weighted sum of visual feature of each image path for the panoramic image scene. v_t is calculated using the attention [7] between the history h_{t-1} and the image patches $\{f_{t,j}\}_{j=1}^{m}$:

$$\begin{aligned} v_t &= \text{Attention}(h_{t-1}, \{f_{t,j}\}_{j=1}^{m}) \\ &= \sum_j \text{softmax}(h_{t-1}W_h(f_{t,j}W_f)^T)f_{t,j}, \end{aligned} \tag{5}$$

where W_h and W_f are learnable projection matrics. The above equation of v_t is for panoramic scene with m viewpoints. APS also supports the navigator which uses visuomotor view as input (e.g., Seq2Seq [3]) and the single visual feature f_t is seen as v_t directly.

Finally, APS decides which action to step based on the history h_t and action embedding u:

$$a_t = \text{softmax}(h_t W_c(u_k W_u^T)), \tag{6}$$

where u_k is the action embedding of the k-th navigable direction. W_c and W_u are learnable projection matrics.

3.4 Adversarial Training of APS

After each unrolling of APS, we comprise the navigation history $\{a_t\}_{t=1}^{T}$ and $\{f_{t,j}\}_{j=1}^{m}$ to obtain the path P. To be consistent with the original training data

Algorithm 1 Training Process of Adversarial Path Sampler

```
NAV: the target navigation model
Speaker: the back-translated instruction model
APS: the adversarial path sampler
$\text{aug}_{\text{aps}}$: collected APS-sampled augmented data

Pre-train NAV with original training set
Pre-train Speaker with original navigation path
Initialize APS
$\text{aug}_{\text{aps}} \leftarrow \varnothing$

while DO_APS do
    $P = \{(f_1, a_1), (f_2, a_2), ..., (f_L, a_L)\} \leftarrow$ APS samples
    $I \leftarrow$ back-translated by Speaker with $P$
    $\mathcal{L}_{\text{NAV}} \leftarrow$ student-forcing loss of NAV using ($P$, $I$)

    Update NAV by minimizing $\mathcal{L}_{\text{NAV}}$
    Update APS by maximizing $\mathcal{L}_{\text{NAV}}$ using Policy Gradient
    $\text{aug}_{\text{aps}} \leftarrow \text{aug}_{\text{aps}} \cup (P, I)$
end while

Train NAV with $\text{aug}_{\text{aps}}$
Fine-tune NAV with original training set
```

whose navigation paths are all shortest paths [3], we transform the sampled paths by APS into shortest paths[1] (same start and end nodes as in the sampled paths). Then we employ the Speaker model [11] to produce one instruction I for each sampled path P, and eventually obtain a set of new augmented pairs (P, I). We train the navigation model (NAV) with (P, I) using student-forcing [3]. The training loss ($\mathcal{L}_{\text{NAV}}$) can be seen as an indicator of NAV's performance under (P, I): the higher $\mathcal{L}_{\text{NAV}}$ is, the worse NAV performs. Hence, in order to create increasingly challenging paths to improve the navigation policy, we define the loss function $\mathcal{L}_{\text{APS}}$ of APS as:

$$\mathcal{L}_{\text{APS}} = -\mathbb{E}_{p(\text{P};\pi_{\text{APS}})}\mathcal{L}_{\text{NAV}}. \tag{7}$$

Since the path sampling process is not differentiable, we adopt policy gradient [30] and view $\mathcal{L}_{\text{NAV}}$ as the reward R to optimize the APS objective. According

[1] Note that transforming the sampled paths into shortest paths can only be done under seen environments. For pre-exploration under unseen environments, we directly use the sampled paths because the shortest path planner should not be exploited in unseen environments.

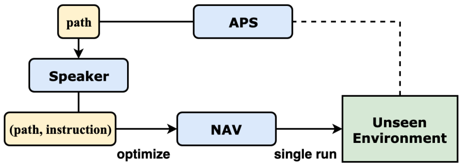


Fig. 4. The optimization flow of environment-based pre-exploration under unseen environments. APS samples paths from the unseen environment to optimize NAV and make it more adaptive. Then, NAV runs each instruction in a single turn.

to the REINFORCE algorithm [34], the gradient is computed as following:

$$\begin{aligned}\nabla_{\pi_{\mathrm{APS}}}\mathcal{L}_{\mathrm{APS}} &\approx -\sum_{t=1}^{T}[\nabla_{\pi_{\mathrm{APS}}}\log p(a_t|a_{1:t-1};\pi_{\mathrm{APS}})R] \\ &\approx -\sum_{t=1}^{T}[\nabla_{\pi_{\mathrm{APS}}}\log p(a_t|a_{1:t-1};\pi_{\mathrm{APS}})(R-b)],\end{aligned} \tag{8}$$

where b is the baseline estimation to reduce the variance and we treat b as the mean of all previous losses. Note that APS is model-agnostic and can be easily integrated into different navigation models, since it only considers the training loss from a navigation model regardless of its model architecture.

Algorithm 1 illustrates the training process of APS. APS aims at maximizing the navigation loss $\mathcal{L}_{\mathrm{NAV}}$ of NAV to create more challenging paths, while NAV tries to minimize $\mathcal{L}_{\mathrm{NAV}}$ to do better navigation:

$$\min_{\mathrm{NAV}}\max_{\mathrm{APS}}\mathcal{L}_{\mathrm{NAV}}. \tag{9}$$

After collecting the challenging paths augmented by APS, we train NAV on them and finally fine-tune NAV with the original training set. The detailed analysis of APS-sampled augmented data is shown in Sec. 4.3.

3.5 Environment-based Pre-Exploration

Pre-exploration is a technique that adapts the navigation model to unseen environments. The navigator can explore the unfamiliar environment first and increase the chance to carry out the navigation instructions under unseen environments. For previous pre-exploration methods like beam search [11] or self-imitation learning (SIL) [32], they are instruction-based which optimizes for each instruction. This will make the navigation path excessive long since it first runs many different paths and then selects the possible best one.

In the real world, when we deploy a robot into a new environment, it might pre-explore and get familiar with the environment, and then efficiently execute the tasks following natural language instructions within this environment. So unlike previous approaches [32,31] that either optimize the given instructions or assume access to all the unseen environments at once, we propose to use our APS method to do the environment-based pre-exploration where the agent pre-explore an environment only for the tasks within the same environment with no

	Test (VLN Challenge Leaderboard)			
Model	NE $\downarrow$	OSR $\uparrow$	SR $\uparrow$	SPL $\uparrow$
Seq2Seq [3]	7.9	26.6	20.4	18.0
+ aug_{rand}	7.8	26.2	21.0	18.8
+ aug_{aps}	**7.5**	**30.1**	**22.5**	**19.3**
+ aug_{aps}+pre-exploration	**6.7**	29.4	**23.2**	**20.8**
Speaker-Follower [11]	7.0	41.2	30.9	24.0
+ aug_{rand}	6.6	43.4	34.8	29.2
+ aug_{aps}	**6.5**	**44.2**	**36.1**	28.8
+ aug_{aps}+pre-exploration	**5.9**	**46.4**	**37.6**	**32.4**
RCM [32]	6.7	43.5	35.9	33.1
+ aug_{rand}	5.9	52.4	44.5	40.8
+ aug_{aps}	**5.8**	**53.9**	**45.1**	**40.9**
+ aug_{aps}+pre-exploration	**5.5**	**55.6**	**45.9**	**40.9**

Table 1. R2R results for Seq2Seq, Speaker-Follower, and RCM under testing set. Models are trained without augmented data, with randomly-sampled augmented path (aug_{rand}), with APS-sampled augmented path (aug_{aps}), and under pre-exploration in unseen environments. Note that those results are run in single turn and with greedy action decoding.

prior knowledge of it. Under an unseen environment, we adopt APS to sample multiple paths (P') and generate the instructions (I') of the sampled paths[2] with the Speaker model [11]. We then use (P', I') to optimize NAV to adapt to the unseen environment as illustrated in Fig. 4. Note that during pre-exploration, we only optimize NAV and let APS fixed[3]. We also present a detailed analysis of our proposed environment-based pre-exploration method in Sec. 4.3.

4 Experiments

4.1 Experimental Setup

R2R Dataset We evaluate the proposed method on the Room-to-Room (R2R) dataset [3] for vision-and-language navigation. R2R is built upon the Matterport3D [4], which contains 90 different environments that are split into 61 for training and validation-seen, 11 for validation-unseen, and 18 for testing sets. There are 7,189 paths and each path has 3 human-written instructions. The validation-seen set shares the same environments with the training set. In contrast, both the validation-unseen and the testing sets contain distinct environments that do not appear during training.

[2] Note that the shortest-path information is not used during pre-exploration.

[3] We have tried to update APS simultaneously with NAV during pre-exploration, but it turns out that under a previous unseen environment without any regularization of human-annotated paths, APS tends to sample too difficult paths to accomplish, *e.g.*, back and forth or cycles. However, those paths will not improve NAV and may even hurt the performance. To avoid this kind of dilemma, we keep APS fixed under the pre-exploration.

	Val-Seen				**Val-Unseen**			
Model	NE ↓	OSR ↑	SR ↑	SPL ↑	NE ↓	OSR ↑	SR ↑	SPL ↑
Seq2Seq [3]	6.0	51.7	39.4	33.8	7.8	27.7	22.1	19.1
+ aug_{rand}	5.3	58.1	43.7	37.2	7.7	28.9	22.6	19.9
+ aug_{aps}	**5.0**	**60.8**	**48.2**	**40.1**	**7.1**	**32.7**	**24.2**	**20.4**
+ aug_{aps}+pre-exploration	-				**6.6**	**37.8**	**27.0**	**24.6**
Speaker-Follower [11]	5.0	61.6	51.7	44.4	6.9	40.7	29.9	21.0
+ aug_{rand}	3.7	74.2	66.4	59.8	6.6	46.6	36.1	28.8
+ aug_{aps}	**3.3**	**74.9**	**68.2**	**62.5**	**6.1**	**46.7**	**38.8**	**32.1**
+ aug_{aps}+pre-exploration	-				**5.2**	**49.1**	**42.0**	**35.7**
RCM [32]	5.7	53.8	47.0	44.3	6.8	43.0	35.0	31.4
+ aug_{rand}	4.1	66.9	61.9	58.6	5.7	52.4	45.6	41.8
+ aug_{aps}	**3.9**	**69.3**	**63.2**	**59.5**	**5.4**	**56.6**	**47.7**	**42.8**
+ aug_{aps}+pre-exploration	-				**5.3**	56.2	**48.0**	**42.8**

Table 2. R2R results for Seq2Seq, Speaker-Follower, and RCM under validation-seen and validation-unseen sets. Models are trained without augmented data, with randomly-sampled augmented path (aug_{rand}), with APS-sampled augmented path (aug_{aps}), and under pre-exploration in unseen environments. Note that those results are run in single turn and with greedy action decoding.

Evaluation Metrics To compare with the existing methods, we report the same used evaluation metrics: Navigation Error (NE), Oracle Success Rate (OSR), Success Rate (SR), and Success Rate weighted by Path Length (SPL). NE is the distance between the agent's final position and goal location. OSR is the success rate at the closest point to the goal that the agent has visited. SR is calculated as the percentage of the final position within 3m from the goal location. SPL, defined in [2], is the success rate weighted by path length which considers both effectiveness and efficiency.

Baselines We experiment with the effectiveness of the model-agnostic APS on 3 kinds of baselines:

- **Seq2Seq** [3], the attention-based seq2seq model that is trained with student forcing (or imitation learning) under the visuomotor view and action space;
- **Speaker-Follower** [11], the compositional model that is trained with student forcing under the panoramic view and action space;
- **RCM** [32], the model that integrates cross-modal matching loss, and is trained using reinforcement learning under the panoramic view and action space.

In the following sections, we use the notations as:

- aug_{rand}: the randomly-sampled augmented path;
- aug_{aps}: the APS-sampled augmented path;
- $\text{model}_{\text{rand}}$: the model trained with aug_{rand};
- $\text{model}_{\text{aps}}$: the model trained with aug_{aps}.

For example, $\text{Speaker-Follower}_{\text{aps}}$ is the Speaker-Follower model trained with the APS-sampled augmented path.

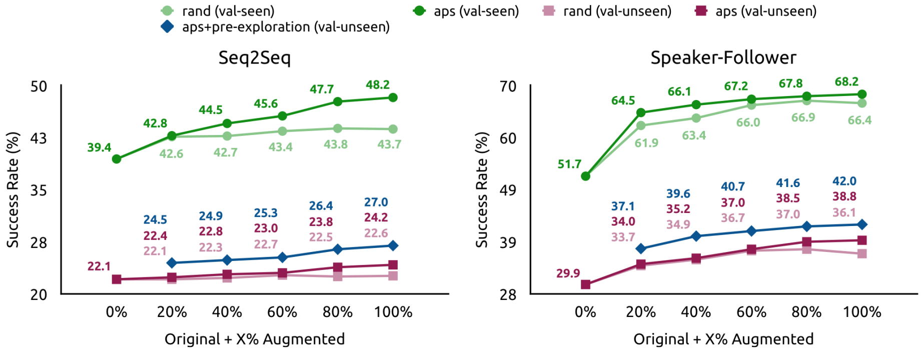


Fig. 5. The comparison between randomly-sampled and APS-sampled under validation-seen and validation-unseen sets for Seq2Seq and Speaker-Follower over different ratios of augmented path used.

For each baseline, we report the results of the model trained without any augmented data, trained with aug_{rand}, and trained with aug_{aps}. For the unseen environments, we also report the results under the pre-exploration.

Implementation Details To follow the previous studies [3,11,32], we adopt ResNet-152 [15] to extract visual features (2048d) for all scene images without fine-tuning; for the navigation instructions, the pre-trained GloVe embeddings [28] are used for initialization and then fine-tuned with the model training. For baseline models, we apply the same batch size 100, LSTM with 512 hidden units, learning rate 1e-4, RL learning rate 1e-5, and dropout rate 0.5. For our proposed APS, the hidden unit of LSTM is also 512, the action embedding size is 128, and the learning rate is 3e-5. We adopt the learning rate 1e-5 under the pre-exploration for the unseen environments. All models are optimized via Adam optimizer [23] with weight decay 5e-4.

For aug_{rand}, we use the same 17K paths as Speaker-Follower [11]. To compare fairly, APS also adversarially samples the same amounts of paths for data augmentation. The navigation models are first trained using augmented data for 50K iterations and then fine-tuned with original human-written instructions for 20K iterations.

4.2 Quantitative Results

Table. 1 and 2 present the R2R results for Seq2Seq [3], Speaker-Follower [11], and RCM [32] under validation-seen, validation-unseen, and testing sets. All models are trained without augmented data, with aug_{rand}, and with aug_{aps}. First, we can observe that under validation-seen set, $\text{Seq2Seq}_{\text{aps}}$ outperforms $\text{Seq2Seq}_{\text{rand}}$ on all evaluation metrics, *e.g.*, 4.5% absolute improvement on Sucess Rate and 2.9% on SPL. Similar trends can be found for Speaker-Follower and RCM where models trained with APS-sampled paths comprehensively surpass models trained with randomly-sampled paths. Since APS can sample increasingly challenging and custom-made paths for the navigator, APS-sampled paths are more effective

	As Testing Set		
Model	train	aug$_{\text{rand}}$	aug$_{\text{aps}}$
Seq2Seq	71.3	20.3	17.7
Seq2Seq$_{\text{rand}}$	**81.4**	26.4	23.8
Seq2Seq$_{\text{aps}}$	78.5	**27.3**	**24.8**

	As Testing Set	
Model	aug$_{\text{rand}}$	aug$_{\text{aps}}$
RCM$_{\text{rand}}$	33.3	31.1
RCM$_{\text{aps}}$	**38.9**	**37.9**

Fig. 6. The success rate under training, randomly-sampled augmented (aug$_{\text{rand}}$), and APS-sampled augmented (aug$_{\text{aps}}$) sets for Seq2Seq and RCM.

than randomly-sample paths and bring in larger improvements on all metrics for all navigation models.

For the unseen environments, all models trained with APS consistently outperform model$_{\text{rand}}$ with 1.6%-2.7% success rate under validation-unseen set and 0.6%-1.5% under testing set. The improvement shows that APS-sampled paths are not only helpful under the seen environments, but also strengthens the model's generalizability under the unseen environments. The results under validation-seen, validation-unseen, and testing sets demonstrate that our proposed APS can further improve the baseline models in all terms of visuomotor view, panoramic view, imitation learning, and reinforcement learning.

And under the pre-exploration, all models gain further improvement, especially on SPL for Seq2Seq and Speaker-Follower due to the prior exploration experience which can shorten the navigation path length. For RCM, they adopt reinforcement learning which may increase the path length but still obtain improvement on success rate.

4.3 Ablation Study

Random Path Sampling vs Adversarial Path Sampling To investigate the advantage of APS, we perform a detailed comparison between randomly-sampled and APS-sampled data. Fig. 5 presents the R2R success rate over different ratios of augmented data used for Seq2Seq and Speaker-Follower. The trend line in light color shows that Seq2Seq$_{\text{rand}}$ cannot gain additional improvement when using more than 60% augmented data. However, for our proposed APS, the sampled augmented path can keep benefiting the model when more data used and achieve 4.5% and 1.6% improvement under validation-seen and validation-unseen sets, respectively. Since aug$_{\text{rand}}$ is sampled in advance, the help to the model is limited. While on the other hand, our proposed APS adversarially learns to sample challenging paths that force the navigator to keep improving. A similar trend can be found for Speaker-Follower where the improvement of Speaker-Follower$_{\text{rand}}$ is also stuck but Speaker-Follower$_{\text{aps}}$ can lead to even better performance.

Difficulty and Usefulness of the APS-sampled Paths For a more intuitive view of the difficulty and usefulness of the APS-sampled paths, we conduct experiments shown in Table 6 to quantitatively compare them with randomly-sample paths. As you can see, the APS-sampled paths seem to be the most

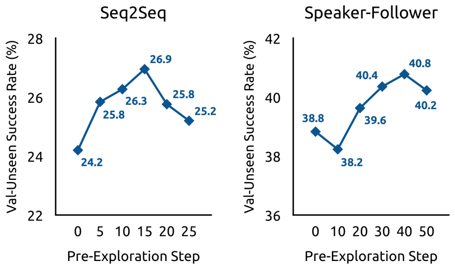


Fig. 7. The success rate under validation-unseen set under different pre-exploration steps for Seq2Seq and Speaker-Follower.

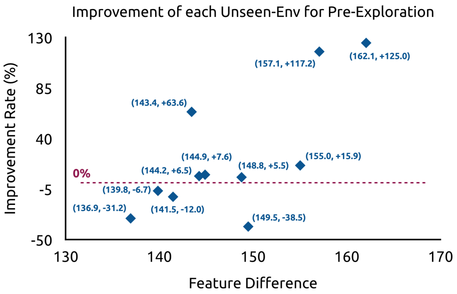


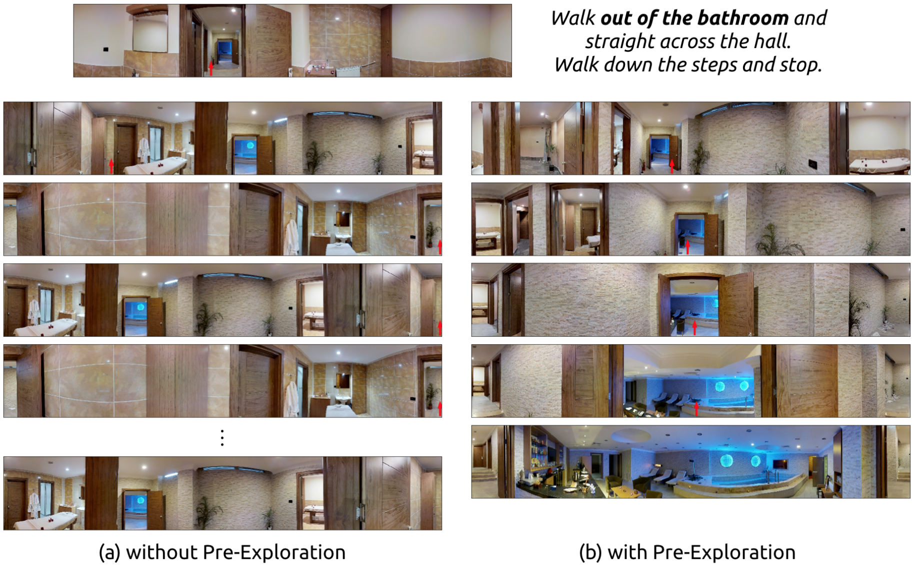
Fig. 8. The improvement of success rate over the scene feature difference under each validation-unseen environment under the pre-exploration. Each point represents a distinct validation-unseen environment.

challenging as all models perform worst on them. These paths can in turn help train a more robust navigation model (Seq2Seq$_{\text{aps}}$) that outperforms the model trained with randomly sampled paths. Moreover, Seq2Seq$_{\text{aps}}$ even performs better on aug$_{\text{rand}}$ than Seq2Seq$_{\text{rand}}$ which shows that aug$_{\text{aps}}$ is not only challenging but also covers useful paths over aug$_{\text{rand}}$.

Pre-Exploration Table. 2 has shown the improvement brought from the pre-exploration. While, those paths in training, validation, and testing sets are all shortest path but the paths sampled from our APS under unseen environments are not promised to be the shortest. With more pre-exploration steps, the model has more opportunities to explore the unseen environment but at the same time, those too complicated paths sampled from APS may hurt the model. Fig. 7 presents the success rate under different pre-exploration steps. It shows a trade-off between the model performance and the iterations of the pre-exploration. For Seq2Seq, 15 steps of pre-exploration come out the best result and 40 steps are most suitable for Speaker-Follower.

We also analyze the performance under the pre-exploration under each unseen environments. Fig. 8 demonstrates the improvement of the success rate over the scene feature difference. Each point represents a distinct validation-unseen environment. The feature difference under each unseen environment is calculated as the mean of the L2-distance between the visual feature of all scenes from that environment and all scenes in the training environments. In general, most of the unseen environments gain improvement under the pre-exploration. We also find a trend that under the environment which has a larger feature difference, it can improve more under the pre-exploration. It shows that under more different environments, the pre-exploration can be more powerful which makes it practical to be more adaptive and generalized to real-life unseen environments.

Qualitative Results Fig. 9 demonstrates the visualization results of the navigation path without and with pre-exploration for the instruction *"Walk out of the bathroom"*. Under the unseen environment, it is difficult to find out a path


*Walk **out of the bathroom** and straight across the hall. Walk down the steps and stop.*

(a) without Pre-Exploration

(b) with Pre-Exploration

Fig. 9. The visualization example of the comparison between without and with the pre-exploration under the validation-unseen environment.

to get out of the unfamiliar bathroom, and as is shown in Fig. 9(a), the model without pre-exploration is stuck inside. In contrast, with the knowledge learned during the pre-exploration phase, the model can successfully walk out of the bathroom and eventually achieve the final goal.

5 Conclusion

In this paper, we integrate counterfactual thinking into the vision-and-language navigation (VLN) task to solve the data scarcity problem. We realize counterfactual thinking via adversarial learning where we introduce an adversarial path sampler (APS) to only consider useful counterfactual conditions. The proposed APS is model-agnostic and proven effective in producing challenging but useful paths to boost the performances of different VLN models. Due to the power of reasoning, counterfactual thinking has gradually received attention in different fields. We believe that our adversarial training method is an effective solution to realize counterfactual thinking in general, which can possibly benefit more tasks.

Acknowledgments. Research was sponsored by the U.S. Army Research Office and was accomplished under Contract Number W911NF-19-D-0001 for the Institute for Collaborative Biotechnologies. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

1. Agmon, N.: Robotic Strategic Behavior in Adversarial Environments. In: IJCAI (2017) 4
2. Anderson, P., Chang, A., Chaplot, D.S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., Zamir, A.R.: On Evaluation of Embodied Navigation Agents. In: arXiv:1807.06757 (2018) 10
3. Anderson, P., Wu, Q., Teney, D., Bruce, J., Johnson, M., Sünderhauf, N., Reid, I., Gould, S., van den Hengel, A.: Vision-and-Language Navigation: Interpreting Visually-Grounded Navigation Instructions in Real Environments. In: CVPR (2018) 1, 2, 3, 5, 6, 7, 9, 10, 11
4. Angel, C., Angela, D., Thomas, F., Maciej, H., Matthias, N., Manolis, S., Shuran, S., Andy, Z., Yinda, Z.: Matterport3D: Learning from RGB-D Data in Indoor Environments. In: 3DV (2017) 9
5. Antoniou, A., Storkey, A., Edwards, H.: Data Augmentation Generative Adversarial Networks. In: 1711.04340 (2017) 4
6. Ashual, O., Wolf, L.: Specifying Object Attributes and Relations in Interactive Scene Generation. In: ICCV (2019) 4
7. Bahdanau, D., Cho, K., Bengio, Y.: Neural Machine Translation by Jointly Learning to Align and Translate. In: ICLR (2015) 6
8. Chen, H., Suhr, A., Misra, D., Snavely, N., Artzi, Y.: Touchdown: Natural Language Navigation and Spatial Reasoning in Visual Street Environments. In: CVPR (2019) 1, 3
9. Chen, L., Zhang, H., Xiao, J., He, X., Pu, S., Chang, S.F.: Counterfactual Critic Multi-Agent Training for Scene Graph Generation. In: ICCV (2019) 4
10. Chou, C.J., Chien, J.T., Chen, H.T.: Self Adversarial Training for Human Pose Estimation. In: APSIPA (2018) 4
11. Fried, D., Hu, R., Cirik, V., Rohrbach, A., Andreas, J., Morency, L.P., Berg-Kirkpatrick, T., Saenko, K., Klein, D., Darrell, T.: Speaker-Follower Models for Vision-and-Language Navigation. In: NeurIPS (2018) 2, 3, 4, 5, 7, 8, 9, 10, 11
12. Garg, S., Perot, V., Limtiaco, N., Taly, A., Chi, E.H., Beutel, A.: Counterfactual Fairness in Text Classification through Robustness. In: AIES (2019) 2, 4
13. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Networks. In: NeurIPS (2014) 4
14. Goyal, Y., Wu, Z., Ernst, J., Batra, D., Parikh, D., Lee, S.: Counterfactual Visual Explanations. In: ICML (2019) 4
15. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: CVPR (2016) 11
16. Hemachandra, S., Duvallet, F., Howard, T.M., adnd Anthony Stentz, N.R., Walter, M.R.: Learning Models for Following Natural Language Directions in Unknown Environments. In: ICRA (2015) 3
17. Hochreiter, S., Schmidhuber, J.: Long Short-term Memory. In: Neural Computation (1997) 6
18. Hong, Z.W., Fu, T.J., Shann, T.Y., Chang, Y.H., Lee, C.Y.: Adversarial Active Exploration for Inverse Dynamics Model Learning. In: CoRL (2019) 4
19. Huang, H., Jain, V., Mehta, H., Baldridge, J., Ie, E.: Multi-modal Discriminative Model for Vision-and-Language Navigation. In: NAACL Workshop (2019) 2, 3
20. Huang, H., Jain, V., Mehta, H., Ku, A., Magalhaes, G., Baldridge, J., Ie, E.: Transferable Representation Learning in Vision-and-Language Navigation. In: ICCV (2019) 3

21. ihan Jain, Magalhaes, G., Ku, A., Vaswani, A., Ie, E., Baldridge, J.: Stay on the Path: Instruction Fidelity in Vision-and-Language Navigation. In: arXiv:1905.12255 (2019) 3
22. Ke, L., Li1, X., Bisk, Y., Holtzman, A., Gan, Z., Liu, J., Gao, J., Choi, Y., Srinivasa, S.: Tactical Rewind: Self-Correction via Backtracking in Vision-and-Language Navigation. In: CVPR (2019) 3
23. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. In: ICLR (2015) 11
24. Kusner, M., Loftus, J., Russell, C., Silva, R.: Counterfactual Fairness. In: NeurIPS (2017) 2, 4
25. Ma, C.Y., Lu, J., Wu, Z., AlRegib, G., Kira, Z., Socher, R., Xiong, C.: Self-Monitoring Navigation Agent via Auxiliary Progress Estimation. In: ICLR (2019) 3
26. Ma, C.Y., Wu, Z., AlRegib, G., Xiong, C., Kira, Z.: The Regretful Agent: Heuristic-Aided Navigation through Progress Estimation. In: CVPR (2019) 3
27. Miyato, T., Dai, A.M., Goodfellow, I.: Adversarial Training Methods for Semi-Supervised Text Classification. In: ICLR (2017) 4
28. Pennington, J., Socher, R., Manning, C.: Glove: Global Vectors for Word Representation. In: EMNLP (2014) 11
29. Roese, N.J.: Counterfactual thinking. In: Psychological Bulletin (1997) 2
30. Sutton, R.S., McAllester, D., Singh, S., Mansour, Y.: Policy Gradient Methods for Reinforcement Learning with Function Approximation. In: NeurIPS (2000) 7
31. Tan, H., Yu, L., Bansal, M.: Learning to Navigate Unseen Environments: Back Translation with Environmental Dropout. In: NAACL (2019) 3, 4, 8
32. Wang, X., Huang, Q., Celikyilmaz, A., Gao, J., Shen, D., Wang, Y.F., Wang, W.Y., Zhang, L.: Reinforced Cross-Modal Matching and Self-Supervised Imitation Learning for Vision-Language Navigation. In: CVPR (2019) 3, 4, 5, 8, 9, 10, 11
33. Wang, X., Xiong, W., Wang, H., Wang, W.Y.: Look Before You Leap: Bridging Model-Free and Model-Based Reinforcement Learning for Planned-Ahead Vision-and-Language Navigation. In: ECCV (2018) 3
34. Williams, R.J.: Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. In: Machine Learning (1992) 8
35. Wu, Y., Bamman, D., Russell, S.: Adversarial Training for Relation Extraction. In: EMNLP (2017) 4
36. Zhang, R., Che, T., Ghahramani, Z., Bengio, Y., Song, Y.: MetaGAN: An Adversarial Approach to Few-Shot Learning. In: NeurIPS (2018) 4
37. Zmigrod, R., Mielke, S.J., Wallach, H., Cotterell, R.: Counterfactual Data Augmentation for Mitigating Gender Stereotypes in Languages with Rich Morphology. In: ACL (2019) 4