Adversarial Reinforced Instruction Attacker for Robust Vision-Language Navigation

Bingqian Lin, Yi Zhu, Yanxin Long, Xiaodan Liang[†], Qixiang Ye, Liang Lin

Abstract—Language instruction plays an essential role in the natural language grounded navigation tasks. However, navigators trained with limited human-annotated instructions may have difficulties in accurately capturing key information from the complicated instruction at different timesteps, leading to poor navigation performance. In this paper, we exploit to train a more robust navigator which is capable of dynamically extracting crucial factors from the long instruction, by using an adversarial attacking paradigm. Specifically, we propose a Dynamic Reinforced Instruction Attacker (DR-Attacker), which learns to mislead the navigator to move to the wrong target by destroying the most instructive information in instructions at different timesteps. By formulating the perturbation generation as a Markov Decision Process, DR-Attacker is optimized by the reinforcement learning algorithm to generate perturbed instructions sequentially during the navigation, according to a learnable attack score. Then, the perturbed instructions, which serve as hard samples, are used for improving the robustness of the navigator with an effective adversarial training strategy and an auxiliary self-supervised reasoning task. Experimental results on both Vision-and-Language Navigation (VLN) and Navigation from Dialog History (NDH) tasks show the superiority of our proposed method over state-of-the-art methods. Moreover, the visualization analysis shows the effectiveness of the proposed DR-Attacker, which can successfully attack crucial information in the instructions at different timesteps.

Index Terms—Vision-and-language navigation, adversarial attack, reinforcement learning, self-supervised learning

A INTRODUCTION

NATURAL language grounded visual navigation task asks an embodied agent to navigate to a goal position following language instructions [1], [2], [3], [4], [5]. It has raised widely research interests in recent years since an instruction-following navigation agent is more flexible and practical in many real-world applications, such as personal assistants and in-home robots. To accomplish successful navigation, the agent needs to extract the key information, e.g., visual objects, specific rooms or navigation directions, from the long instruction according to dynamic visual observation for guiding navigation at each timestep. However, due to the complexity and semantic ambiguity of the natural language, it is hard for the navigators to effectively learn cross-modality alignment and capture accurate semantic intentions from the instruction by training with limited human-annotated instruction-path data.

Prior works mainly employed the data augmentation strategy to solve the data scarcity in navigation tasks [6], [7], [8]. [6] proposed a speaker-follower framework to generate augmented instructions within randomly sampled paths. However, generating a large amount of the whole instructions is at high costs and may not contribute to the emphasis of the most instructive information. [7] and [8] put more focus on creating challenging augmented paths and diverse visual scenes, while generated augmented instructions by employing the speaker model in [6] directly. Therefore, the enhancement of the instruction understanding ability of the navigator might also be limited.

In recent years, there have been increasing attentions in designing the adversarial attacks for natural language processing (NLP) tasks to verify and improve the robustness of NLP models [9], [10], [11], [12]. Inspired by this, we consider the following question: Can we design adversarial attacks on the instruction to generate helpful adversarial samples for improving the robustness of the navigator? A simple way to generate adversarial instructions is to borrow the existing attack methods on NLP [12], [13] tasks directly. However, it is difficult since existing adversarial attacks on NLP are often optimized by some classification-based goal functions [9], [12], which are unreachable in the navigation tasks. Moreover, the key instruction information for navigation changes dynamically while these attack methods developed on NLP are designed in the static setting.

In this paper, we make the first attempt for introducing the adversarial attacks on the language instruction of navigation tasks to improve the robustness of navigators. Specifically, we propose a Dynamic Reinforced Instruction Attacker (DR-Attacker), which learns to minimize the navigation reward by dynamically *destroying* key instruction information and generating perturbed instruction at each timestep. Then, an effective adversarial training strategy is adopted to improve the robustness of the navigator, by asking it to maximize the navigation reward with the perturbed instruction. To encourage the agent to be aware of actual key information and improve the fault-tolerance ability with perturbed instruction, an auxiliary self-supervised reasoning task is also introduced for the navigator, requiring it to distinguish the actual attacked word of the DR-Attacker at each timestep according to the instruction and current visual

- [†] *Xiaodan Liang is the corresponding author.*

- *Bingqian Lin, Yanxin Long, Xiaodan Liang and Liang Lin are with Sun Yat-sen University, Guangzhou, China.*
E-mail:{bingqianlin@126.com,yestinl1129@gmail.com, xdliang328@gmail.com, linliang@ieee.org}
- *Yi Zhu, Qixiang Ye are with University of Chinese Academy of Sciences (UCAS), Beijing, China.*
E-mail: {zhu.yee@outlook.com, qxye@ucas.ac.cn}

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2021.3097435, IEEE Transactions on Pattern Analysis and Machine Intelligence

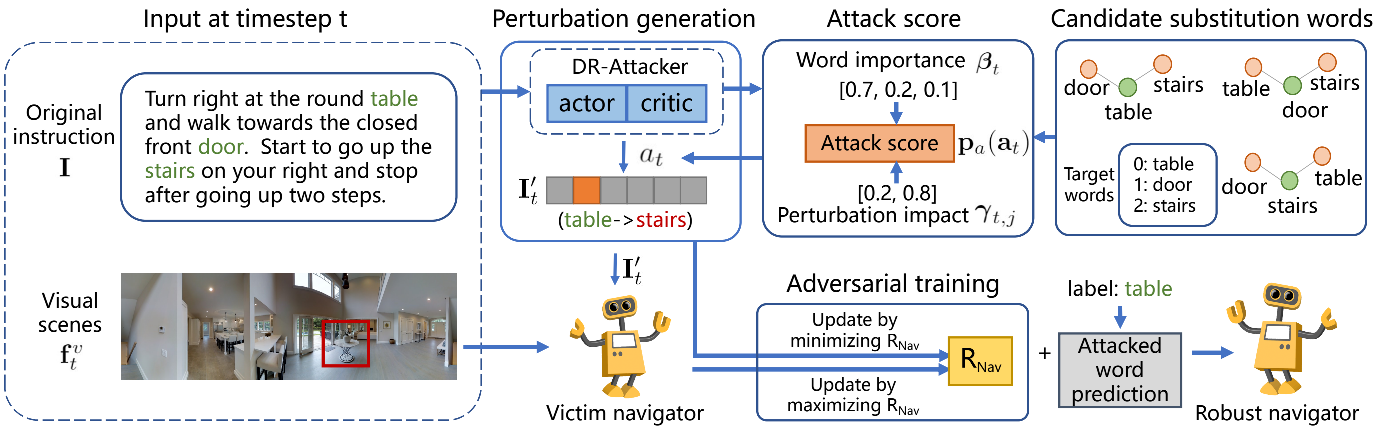


Fig. 1: The overview of our proposed method. At timestep t, the DR-Attacker receives the visual observation and original instruction, and generates the perturbed instruction $\mathbf{I}'_t$ by substituting the selected target word with the best candidate word according to the attack score. The victim navigator, which receives the perturbed instruction, is enforced to maximize the navigation reward R_{Nav} with an adversarial setting and reasoning the actual attacked words by DR-Attacker to enhance the model robustness.

observation. As a result, more accurately the DR-Attacker attacks the important instruction information, more possible that the agent is able to capture the actual key information for navigation.

Since navigation is a sequential decision making problem without direct classification-based objectives, we formulate the perturbation generation as a Markov Decision Process, and present a reinforcement learning (RL) resolution to generate the perturbed instructions by misleading the navigator to move to the wrong target position. At each timestep, the policy agent, i.e., our proposed DR-Attacker, substitutes the most crucial target word in the current instruction with the best candidate substitution word which has maximum perturbation impact, according to a learnable attack score. As a result, the DR-Attacker can learn to highlight the important parts in instructions to generate adversarial samples at different timesteps. To enhance the navigation robustness, the victim navigator, which receives the perturbed instruction, is enforced to be immune to the perturbation under the adversarial setting, as well as correctly reasoning the actual attacked words by the DR-Attacker. The overview of our proposed method is presented in Figure 1. Suppose a person receives the perturbed instruction $\mathbf{I}'_t$ where the word "table" is substituted with the word "stairs". With the good understanding of the instruction and visual environment, he can distinguish the noisy word and still make the correct navigation decision. Therefore, the perturbed instructions, which can be viewed as hard negative samples, can effectively encourage the victim navigator to understand the multi-modality observations and have the self-correction ability thus become more robust.

Experimental results on both Navigation from Dialog History (NDH) and Vision-and-Language Navigation (VLN) show the superiority of the proposed method over other competitors. Moreover, the quantitative and qualitative results show the effectiveness of the proposed DR-Attacker, which causes significant navigation performance drop by only disturbing most crucial instruction information.

The merits of our proposed DR-Attacker are summarized as follows: First, DR-Attacker can generate perturbed instruction dynamically by capturing and destroying key instruction information in different navigation timesteps. Second, DR-Attacker can be optimized via gradient-based methods under the unsupervised setting, by formulating the perturbation generation as a sequential decision making problem. Last but not least, the adversarial samples produced by DR-Attacker are beneficial for improving the model robustness.

The main contributions of this paper are summarized as follows:

- We take the first step to introduce the adversarial attack on the language instruction of navigation tasks to learn robust navigators. Different from existing adversarial attacking paradigm developed on NLP tasks which are generally static, the proposed adversarial attack is dynamic during the navigation process.
- By formulating the perturbation generation as a Markov Decision Process, the proposed instruction attacker, called Dynamic Reinforced Instruction Attacker (DR-Attacker), can be optimized by the reinforcement learning algorithm to achieve effective perturbation, without the need of classification-based objectives.
- To improve the robustness of the navigator, an alternative adversarial training strategy and an auxiliary self-supervised reasoning task are employed to train the navigator on perturbed instructions, which can effectively enhance the cross-modal understanding ability of the navigator.
- Experimental results on two popular natural language grounded visual navigation tasks, i.e., Vision-and-Language Navigation (VLN) and Navigation from Dialog History (NDH) show that the model robustness can be effectively enhanced by the proposed method. Moreover, both the quantitative results and visualized results show the effectiveness of the proposed DR-Attacker.

The remainder of this paper is organized as follows. Section B gives a brief review of the related work. Section C describes the problem setup of natural language grounded visual navigation tasks and then introduces our proposed method. Experimental results are provided in Section D. Section E concludes the paper and presents some outlook for future work.

B Related Work

B.1 Natural Language Grounded Visual Navigation

Natural language grounded visual navigation tasks [1], [2], [3], [4], [5], [14], [15] have attracted extensive research interests in recent years since they are practical and pose great challenges for vision-language understanding tasks [16], [17], [18], [19]. In this paper, we mainly focus on two natural language grounded navigation tasks, namely, Vision-and-Language Navigation (VLN) [1] and Navigation from Dialog History (NDH) [2].

Vision-and-Language Navigation (VLN) [1], [6], [7], [8] was first proposed by [1], where a navigation agent is asked to move to the goal position following the navigation instruction. Specifically, the instruction is a sequence of declarative sentences such as "Walk down stairs. Walk past the chartreuse ottoman in the TV room. Wait in the bathroom door threshold." Therefore, to successfully navigate to the goal position, the agent needs to understand the instruction well and learn to ground the instruction to visual observations. To achieve this, [20] proposed Reinforced Cross-Modal Matching (RCM) approach to enforce cross-modal grounding both locally and globally via reinforcement learning (RL). [21] designed visual-textual co-grounding module to distinguish different instruction parts as the ones have completed and the ones need to complete regarding visual observations. To better encourage the navigator to sufficiently understand the diverse instructions and navigation environments, existing works adopted the data augmentation strategy [6], [7], [8] to solve the data scarcity in the original dataset. A speaker-follower model was proposed by [6] to produce augmented instructions with randomly-sampled paths. [7] proposed Environmental Dropout to create new (environment, path, instruction) triplets while utilizing the speaker model in [6] directly for generating the augmented instructions.

The Cooperative Vision-and-Dialog Navigation (CVDN) dataset was recently proposed by [2] and **Navigation from Dialog history (NDH)** is a task proposed on CVDN dataset, which requires an agent to move towards the goal position following a sequence of dialog history. Although the visual scenes in CVDN dataset are similar to the R2R dataset proposed on VLN task [1], the instruction in the CVDN dataset, which is composed of dialog history and current question-answer pair, is harder for the agent to understand and perform visual grounding since it is longer and more complicated than the instruction on VLN task. To better explore useful textual information for successful navigation, [22] proposed Cross-modal Memory Network (CMN) to exploit the rich information in dialog history. [23] employed a pretraining scheme by using image-text-action triplets for improving the instruction understanding and cross-modality alignment.

While existing methods have achieved some improvements in enhancing the instruction understanding by data augmentation [6], [7], [8] or pretraining [23], [24], the quality of the augmented instructions is rarely noticed, leading to limited improvement of the model robustness. In contrast, we adopt an adversarial attack paradigm to encourage the generation of meaningful adversarial instructions, which can serve as hard augmented samples to better enhance the navigation robustness.

B.2 Adversarial Attacks in NLP

Adversarial attacks have been widely used in the image domain [25], [26], [27], [28], [29] to validate the robustness of the deep neural network models [30], [31]. In recent years, many researchers of NLP fields put their focus on introducing adversarial attacks for the NLP tasks, which can serve as a powerful tool for evaluating the model vulnerability, and more importantly, improving the robustness of NLP models [12], [32], [33], [34], [35]. The key principle of adversarial attacks is to impose imperceptible perturbation by human on the original input while easily fool the neural model to make the incorrect prediction. Most adversarial attacks on NLP tasks are word-level attacks [12], [13] or character-level attacks [9], [11]. HotFlip [36] introduced white-box adversarial samples based on an atomic flip operation to trick a character-level neural classifier. [13] proposed a word-level attack model based on sememe-based word substitution method and particle swarm optimization-based search algorithm, which was implemented on Bi-LSTM [37] and BERT [38]. Due to the discrete characteristic of the natural language, the imposed adversarial attacks on the language, such as inserting, removing or replacing a specific character or word, can easily change the meaning or break the grammaticality and naturality of the original sentence [13], [39]. Therefore, the adversarial perturbation on the language is essentially easy to be perceived by human rather than that in image.

Our introduced attack on the instruction can be viewed as an adversarial attack naturally due to the following aspects. First, we constraint our DR-Attacker to replace a single word at a specific timestep to control the magnitude of the perturbation to be small enough. Second, although the local key information, e.g., a visual object word is destroyed, the human, which is able to comprehend the long-term intention of the instruction and reasoning original instruction information according to the current visual observation, cannot be misled easily by such perturbation. However, the agent, which tends to learn the simple alignment of the instruction and visual observation, is more easily to be misled and to get stuck. Third, the replacement is conducted between words belonging to the same characteristic, ensuring the grammaticality and naturality of the original sentence. Since incorrect visual object, location or action words in an instruction is easy to appear in realistic scenes, e.g., a wrong annotation by human or an object previously existing but disappearing in the original scene, we impose the perturbation on visual object or location words rather than uninformative words, which can be more beneficial for enhancing the navigation robustness.

In contrast to existing adversarial attacks on NLP which are generally static and optimized with classification-based objectives, our proposed DR-Attacker can generate dynamic perturbation on the instruction, and can be optimized by the RL paradigm under the unsupervised setting. Like other existing works which train the models on the perturbed training samples to improve the robustness of NLP models [33], [40], [41], [42], we also develop the adversarial training strategy to improve the robustness of the navigator using the perturbed instructions generated at each timestep. Moreover, we introduce an auxiliary self-supervised reasoning task during the adversarial training stage, which can better promote the adversarial training results.

B.3 Adversarial Attacks in Navigation

Although adversarial attacks are popular in verifying and improving the robustness of the deep learning models in both image [25], [26], [27], [28] and NLP [9], [10], [11], [12], [13], [32], [33], [34], [35], [43] domains, there are few works attempting to employ the adversarial attacks for improving the robustness of the embodied navigation agents, since the setting and environment in navigation is usually dynamic and complex. [44] took the first attempt to introduce spatio-temporal perturbations on the visual objects for embodied question answering (EQA) task [45], by perturbing the physical properties (e.g., texture or shape) of visual objects. They used the available ground-truth labels to guide the perturbation generation by using classification-based objectives. Compared with the collection of diverse visual environments to improve the robustness of the agent, annotating large-amount of high-quality and informative instruction is more difficult and labor-intensive for the natural language grounded visual navigation task. Therefore, in contrast to [45], we make the first attempt to introduce adversarial attacks for the existing available instruction data in this paper, to mitigate the scarcity of available high-quality instructions which largely limits the navigation performance of existing instruction-following agents. Moreover, our introduced perturbation can be optimized in an unsupervised way, which is more practical.

B.4 Automatic Data Augmentation

Automatic data augmentation aims to learn data augmentation strategies automatically according to the target model performance instead of designing augmentation strategies manually based on the expertise knowledge. AutoAugment [46] formulates the automatic augmentation policy search as a discrete search problem and employs a reinforcement learning (RL) framework to search the policy consisting of possible augmentation operations. However, high computational cost is required for training and evaluating thousands of sampled policies in the search process. To speed up policy search, many variants of AutoAugment are proposed [42], [47], [48], [49], [50]. PBA [47] introduces population-based training to efficiently train the network parallelly across different CPUs or GPUs. Fast AutoAugment [48] moves the costly search stage from training to evaluation through bayesian optimization. Adversarial AutoAugment [42] directly learns augmentation policies on target tasks and develops an adversarial framework to jointly optimize target network training and augmentation policy search. The most related work to our proposed method is Adversarial AutoAugment [42], where the policy sampler and the target model are jointly optimized in an adversarial way. The difference between our method and Adversarial AutoAugment is that our augmented samples are generated through the adversarial attack rather than the composition of augmentation strategies, which is constrained to be small in magnitude while impact the agent performance largely.

C Method

In this section, we describe the natural language grounded visual navigation task first and then introduce our proposed method. The problem setup is given in Sec. C.1. The details of our proposed Dynamic Reinforced Instruction Attacker (DR-Attacker), including the optimization of the perturbation generation, the adversarial training with the auxiliary self-supervised reasoning task, and the model details are presented in Sec. C.2.

C.1 Problem Setup

Natural language grounded visual navigation task requires a navigator to find a route (a sequence of viewpoints) from a start viewpoint to the target viewpoint following the given instruction $\mathbf{I}$. For the NDH task, the instruction $\mathbf{I}$ is composed of $< t_0, \mathbf{Q}_1, \mathbf{H}_1, \mathbf{Q}_2, \mathbf{H}_2, ..., \mathbf{Q}_t, \mathbf{H}_t >$, which includes the given target object t_0, the questions $\mathbf{Q}$ and the answers $\mathbf{H}$ till the turn t ($0 \leq t \leq T$, where T is the total number of question-answer turns from the intial position to the target room). For the VLN task, the instruction $\mathbf{I}$ is composed of $< \mathbf{G}_1, \mathbf{G}_2, ..., \mathbf{G}_M >$, where $\mathbf{G}_m$ ($1 \leq m \leq M$) denotes a single sentence and M denotes the number of sentences. Since the t_0, $\mathbf{Q}_t$, $\mathbf{H}_t$, $\mathbf{G}_m$ can all be represented by word tokens, for both NDH and VLN tasks, we formulate the instruction $\mathbf{I}$ as a set of word tokens, $\mathbf{I} = \{w_0, ..., w_L\}$, where L is the length of the instruction. At timestep t, the navigator receives a panoramic view as the visual observation. Each panoramic view is divided into 36 image views $\{o_{t,i}\}_{i=1}^{36}$, with each of views $o_{t,i}$ containing a RGB image $b_{t,i}$ accompanied with its orientation ($\theta_{t,i}^1$,$\theta_{t,i}^2$), where $\theta_{t,i}^1$ and $\theta_{t,i}^2$ are the angles of heading and elevation, respectively. We follow the [7] to obtain the view feature $\mathbf{f}_{t,i}^v$. Regarding the visual observations and instructions, the navigator infers the action for each step t from the candidate actions list, which consists of J neighbours of the current node in the navigation graph and a stop action. Generally, the navigator is a sequence-to-sequence model with the encoder-decoder architecture [1], [2].

C.2 Dynamic Reinforced Instruction Attacker

C.2.1 Perturbation Generation as an RL Problem

Since there is no direct label as that in the classification-based tasks [9], [12] for judging the success of attack in such navigation tasks, we use a reinforcement learning (RL) framework to formulate the perturbation generation. The framework contains two major components: an environment model $\mathbf{E}_\mu$ which is a well-trained navigator (also called as victim navigator), and an instruction attacker π_ϕ, which can be viewed as the policy agent. The attacker π_ϕ learns to disturb the correct action decision of $\mathbf{E}_\mu$ by generating perturbed instruction $\mathbf{I}_t'$ for $\mathbf{E}_\mu$ at each timestep t. μ and ϕ denote the parameters of the environment model and attacker, respectively. Under the RL setting, the state $s_t \in S$ is the visual state $\mathbf{f}_t^v$. The action $a_t \in A$ is the perturbation operation by substituting the selected target word in the original instruction with a candidate word. The construction details of the target word set and candidate substitution word set for each instruction are given in Sec. C.2.3. Note that the attack operation is sequentially conducted at each navigation step t rather than once at the beginning since the key instruction information changes dynamically during the navigation process.

To measure the success of the attack and design reasonable reward for optimizing the attacker in such navigation tasks, we propose "deviation from the target position" as a metric. That is, the goal of the attacker is to enforce the navigator to make the wrong navigation trajectory and stop at a position which is far from the target position. Therefore, the reward r_t will be negative for the attacker if the victim navigator stops within Z meters around the target viewpoint at the final step, otherwise the reward will be positive. Z is a predefined distance threshold. We also adopt a direct reward [51] at each non-stop step t by considering the progress, i.e., the change of the distance to the target viewpoint made by current timestep. If the navigator makes positive progress to the target position at non-stop step t, the direct reward r_t will be negative. Similar to [7], the reward in our RL setting is set as a predefined constant. To satisfy the 'small perturbation' principle of adversarial samples [9], [12], [13], [33], the attacker is required to substitute only one word in the instruction at each timestep.

Without the loss of generality, we apply the Advantage Actor-Critic (A2C) [52] algorithm to iteratively update the parameters of the attacker π_ϕ. A2C framework contains a policy network $\pi(\mathbf{s}|\phi_\pi)$ (here is the attacker) and a value network $V(\mathbf{s}|\phi_v)$ to learn a optimal policy. ϕ_π and ϕ_v denote the parameters of the network. Given the *state-action-reward* (s_t, a_t, r_t) of $\forall t \in (0, N)$ observation at each step t, the algorithm computes the total accumulated reward R_t, the policy gradient ∇_{pg}, the value gradient ∇_v and the entropy gradient ∇_h by:

$$R_t = \sum\nolimits_{i=t}^{N} \gamma^{i-t} r_t + \gamma^{N-t} V(s_{N+1}), \tag{1}$$

$$\nabla_{pg} = \nabla_{\phi_\pi} \log(\pi(s_t, a_t|\phi_\pi)) A_t, \tag{2}$$

$$\nabla_v = \frac{\partial (V(s_t|\phi_v) - R_t)^2}{\partial \phi_v}, \tag{3}$$

$$\nabla_h = \nabla_{\phi_\pi} \sum\nolimits_{i=0}^{N} \log(\pi(s_i, a_i|\phi_\pi)) \pi(s_i, a_i|\phi_\pi). \tag{4}$$

where $\gamma \in [0, 1)$ is the discount factor. $A_t = R_t - V(s_t)$ is the advantage. Subsequently, an optimization step is performed in the direction that maximizes both $\mathbb{E}[R_t]$ (direction ∇_{pg}) and the entropy of $\pi(s_t)$ (direction ∇_h), as well as minimizes the mean squared error of $V(s_t)$ (direction -∇_v). Therefore, by using the RL paradigm, the attacker can learn to generate the perturbed instructions at each timestep for disturbing the action decision of the navigator and misleading it to stop at the wrong target position. In our settings, the value network is a two-layer MLP.

C.2.2 Adversarial Training with Auxiliary Self-supervised Task

For improving the navigation robustness, we develop an effective adversarial training strategy, which can encourage the joint optimization for the victim navigator and the attacker. Through alternative optimization under the adversarial setting, the attacker can iteratively learn to create misleading instructions for confusing the victim navigator, while the victim navigator is trained on the perturbed instructions to enhance the model robustness. Motivated by [53], we use the RL strategy for training both the victim navigator and the attacker, and formulate the adversarial setting as the two-player zero-sum Markov games. At each timestep t, both the attacker and the victim navigator receive the visual observation $\mathbf{f}_t^v$ and the language instruction $\mathbf{I}_t$ ($\mathbf{I}_t'$ for the navigator, $\mathbf{I}_t$ is invariant while $\mathbf{I}_t'$ is variable). Then the attacker takes the action by generating the perturbed instruction, and the navigator takes the action by moving to the next viewpoint. With the inverse objective of the navigation, i.e., the navigator is supposed to stop at the nearest point from the target position, an inverse reward is set for the attacker and the navigator: $r_\pi = -r_\eta$ (r_η is represented by R_{Nav} in Figure 1), where π and η represent the policies for the attacker and the victim navigator, respectively. Therefore, our adversarial setting can be represented by:

$$r_\eta^* = \min_\pi \max_\eta r_\eta(\eta, \pi). \tag{5}$$

We conduct the alternative optimization procedure between the navigator and the attacker, namely, keep the parameters of one agent fixed and optimize another. The optimization procedure of adversarial training is given in Algorithm 1. At stage 1, we pre-train the navigator and use the pre-trained navigator to pre-train the attacker. At stage 2, we conduct alternative iteration procedure between the navigator and the attacker to implement the joint optimization. For facilitating implementation, the RL strategy for training the victim navigator also follows the A2C algorithm which was similar to [7].

To encourage the agent to capture actual key information and improve the fault-tolerance ability with perturbed instructions, which is important for robust navigation, we introduce an auxiliary self-supervised reasoning task during the training phases of the victim navigator, by asking the navigator to predict the actual attacked word by the attacker at each timestep t:

$$\mathbf{p}_c(\mathbf{c}) = \mathrm{Softmax}((\mathbf{f}^w \mathbf{W}_e)(\tilde{\mathbf{h}}_t \mathbf{W}_h)^T). \tag{6}$$

where $\mathbf{c}$ is the target word set for the given instruction $\mathbf{I}$ and $\mathbf{p}_c(\mathbf{c})$ denotes the prediction probability. $\mathbf{f}^w \in \mathbb{R}^{L' \times D_w}$ denotes the target word features. L' is the size of the target word set. $\tilde{\mathbf{h}}_t \in \mathbb{R}^{1 \times D_h}$ represents the visual-and-instruction aware hidden state feature of the decoder [7] in the navigator. $\mathbf{W}_e \in \mathbb{R}^{D_w \times D_p}$ and $\mathbf{W}_h \in \mathbb{R}^{D_h \times D_p}$ denote the learnable linear transformations. D_w, D_h and D_p denote the feature dimensions. The prediction is optimized by cross-entropy loss and the ground-truth label is the actual attacked word by the attacker. As a result, the probability that the agent captures the actual important instruction information and haves the self-correction ability can be increased with the

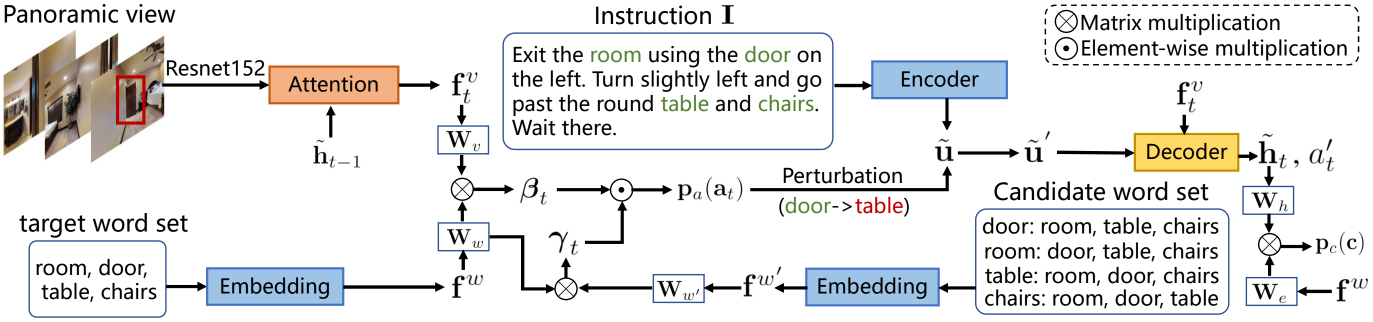


Fig. 2: The forward processes of the DR-Attacker and the navigator. The attack score $\mathbf{p}_a(\mathbf{a}_t)$ is calculated by the element-wise multiplication of the word importance vector $\boldsymbol{\beta}_t$ and the substitution impact matrix $\boldsymbol{\gamma}_t$. After performing the perturbation operation on the original instruction $\tilde{\mathbf{u}}$ to generate perturbed instruction $\tilde{\mathbf{u}}'$, the decoder of the navigator gets the perturbed instruction $\tilde{\mathbf{u}}'$ and the attended visual feature $\mathbf{f}_t^v$ to predict the navigation action a_t' at timestep t. The updated hidden state $\tilde{\mathbf{h}}_t$ of the decoder and the target word feature $\mathbf{f}^w$ are used to calculate the actual attacked word prediction probability $\mathbf{p}_c(\mathbf{c})$.

```
Algorithm 1: Adversarial Training
  Input: the navigator NAV with policy $\eta$, the attacker
         ATT with policy $\pi$
  Output: Optimized parameters $\theta^\eta_{N_{iter}}$ for $\eta$ and $\theta^\pi_{N_{iter}}$
          for $\pi$
  // Stage 1: Initialization
  Pre-train NAV with original training set to get $\theta^\eta_0$
  Pre-train ATT with the pretrained NAV of fixed
    parameters $\theta^\eta_0$ to get $\theta^\pi_0$
  // Stage 2: Adversarial training
  for $i = 1 : N_{iter}$ do
      // Fix $\theta^\pi$ to optimize $\theta^\eta$
      $\theta^\eta_i \leftarrow \theta^\eta_{i-1}$
      for $j = 1 : N_\eta$ do
          $\{(s_t,a^\eta_t,r^\eta_t)\} \leftarrow$ rollout($\mathbf{I}'_t$, $f^v_t$, $\eta_{\theta^\eta_i}$, $\pi_{\theta^\pi_{i-1}}$)
          $\theta^\eta_i \leftarrow$ policyOptmizer($\{(s_t,a^\eta_t, r^\eta_t)\}$, $\eta$, $\theta^\eta_i$)
      end
      // Fix $\theta^\eta$ to optimize $\theta^\pi$
      $\theta^\pi_i \leftarrow \theta^\pi_{i-1}$
      for $j = 1 : N_\pi$ do
          $\{(s_t,a^\pi_t,r^\pi_t)\} \leftarrow$ rollout($\mathbf{I}_t$, $f^v_t$, $\eta_{\theta^\eta_i}$, $\pi_{\theta^\pi_i}$)
          $\theta^\pi_i \leftarrow$ policyOptmizer($\{(s_t, a^\pi_t, r^\pi_t)\}$, $\pi$, $\theta^\pi_i$)
      end
  end
  return $\theta^\eta_{N_{iter}}$, $\theta^\pi_{N_{iter}}$
```

accuracy improvement of the attacker for attacking key instruction information. Therefore, through the auxiliary self-supervision reasoning task, the enhancement of the attacker can effectively lead to the improvement of the navigator.

C.2.3 Model Details

Forward Process of the Instruction Attacker. In this part, we describe the forward process of the proposed DR-Attacker, i.e., the attacker π_ϕ in detail. At each timestep t, the DR-Attacker calculates the action prediction probability, also referred to as the attack score, by considering both the word importance in the current instruction and the substitution impact of different candidate words (illustrated in Figure 1). Within the prior that the words indicate visual object (e.g., "door") and location (e.g., "bathroom") are most informative for guiding the navigation, we construct the target word set by selecting these two kinds of words for each instruction in advance. For target word w_j ($0 \leq j \leq L'$, L' is the size of target word set) in the instruction $\mathbf{I}$, we denote the candidate substitution word set of w_j as $\{w'_{j,k}\}_{k=1}^K$, where K is the size of candidate substitution word set. To promote the understanding of the given instruction as well as keep a reasonable set size, we choose the remained target words in the same instruction to construct the candidate substitution word set for the specific target word. At timestep t, a word importance vector $\boldsymbol{\beta}_t \in \mathbb{R}^{L'}$ is first caculated by:

$$\boldsymbol{\beta}_t = \text{Softmax}((\mathbf{f}^w\mathbf{W}_w)(\mathbf{f}_t^v\mathbf{W}_v)^T), \tag{7}$$

where $\mathbf{f}^w \in \mathbb{R}^{L' \times D_w}$ and $\mathbf{f}_t^v \in \mathbb{R}^{1 \times D_v}$ represent the word features encoded by BiLSTM of target words and attended visual feature [7], respectively. $\mathbf{W}_w \in \mathbb{R}^{D_w \times D_p}$ and $\mathbf{W}_v \in \mathbb{R}^{D_v \times D_p}$ are the learnable linear transformations that convert the different features into the same embedding space. D_w, D_v and D_p represent the feature dimensions. Then, the substitution impact of different candidate words for each target word w_j is obtained by:

$$\boldsymbol{\gamma}_{t,j} = \text{Softmax}((\mathbf{f}_j^w\mathbf{W}_w)(\mathbf{f}_j^{w'}\mathbf{W}_{w'})^T), \tag{8}$$

where $\mathbf{f}_j^w \in \mathbb{R}^{1 \times D_w}$ and $\mathbf{f}_j^{w'} \in \mathbb{R}^{K \times D_w}$ denote the word features of target word w_j and candidate words w'_j. $\mathbf{W}_{w'} \in \mathbb{R}^{D_w \times D_p}$ is the learnable linear transformation. After calculating the substitution impact of different candidate words for all the target words in the instruction to obtain the substitution impact matrix $\boldsymbol{\gamma}_t \in \mathbb{R}^{L' \times K}$, the attack score $\mathbf{p}_a(\mathbf{a}_t) \in \mathbb{R}^{L' \times K}$, i.e., the action prediction probability of the DR-Attacker is calculated by:

$$\mathbf{p}_a(\mathbf{a}_t) = \text{Softmax}(\boldsymbol{\beta}_t \circ \boldsymbol{\gamma}_t). \tag{9}$$

where $\circ$ denotes the element-wise multiplication. $\mathbf{a}_t$ represents the candidate action set with the size of $L' \times K$. Through the learnable attack score $\mathbf{p}_a(\mathbf{a}_t)$, the DR-Attacker can learn to generate the optimal perturbation at each timestep t. Note that while there will be a semantic change compared with the original target word based on our word substitution strategy, we do not distinguish the perturbed instruction with the conventional adversarial samples. This

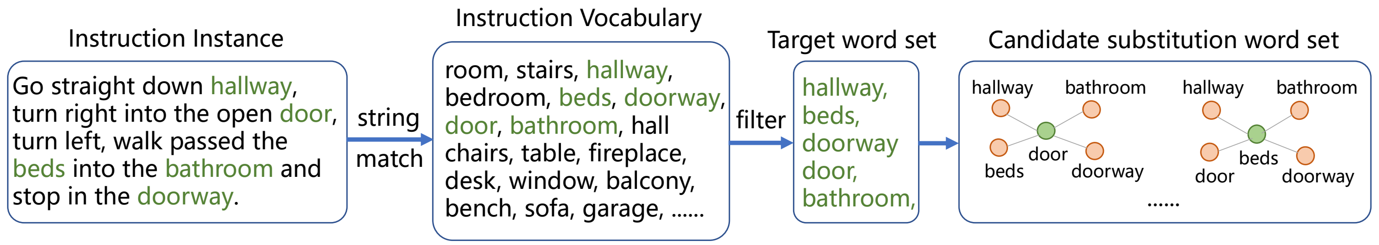

Instruction Instance
Go straight down hallway, turn right into the open door, turn left, walk passed the beds into the bathroom and stop in the doorway.
string match
Instruction Vocabulary
room, stairs, hallway, bedroom, beds, doorway, door, bathroom, hall chairs, table, fireplace, desk, window, balcony, bench, sofa, garage, ...
filter
Target word set
hallway, beds, doorway door, bathroom,
Candidate substitution word set
hallway
bathroom
door
beds
doorway
hallway
bathroom
beds
door
doorway
......

Fig. 3: The construction details of target word set and candidate substitution word set on VLN. The target word set is constructed for each instruction by conducting string match between the instruction and the instruction vocabulary which only contains words indicating visual objects and locations. The candidate substitution word set for each target word is built by collecting the remaining target words in the same instruction.

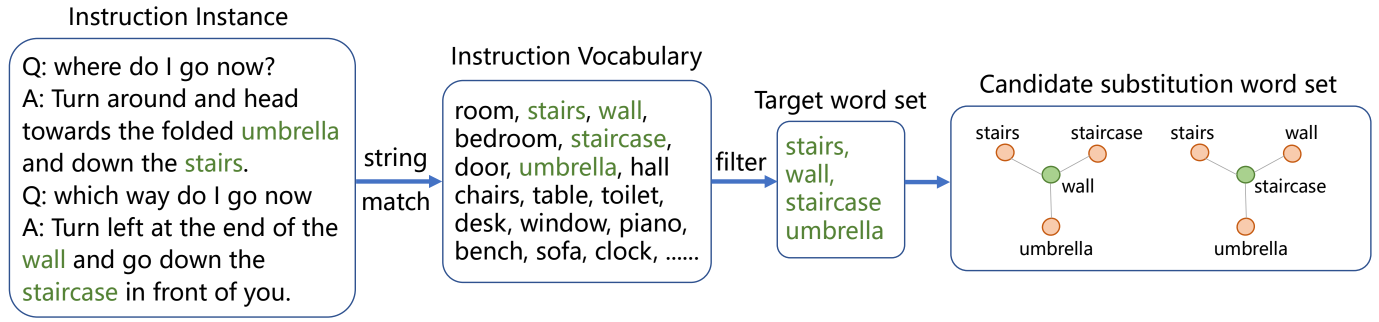

Instruction Instance
Q: where do I go now?
A: Turn around and head towards the folded umbrella and down the stairs.
Q: which way do I go now
A: Turn left at the end of the wall and go down the staircase in front of you.
string match
Instruction Vocabulary
room, stairs, wall, bedroom, staircase, door, umbrella, hall chairs, table, toilet, desk, window, piano, bench, sofa, clock, ...
filter
Target word set
stairs, wall, staircase umbrella
Candidate substitution word set
stairs
staircase
wall
umbrella
stairs
wall
staircase
umbrella

Fig. 4: The construction details of target word set and candidate substitution word set on NDH. Since the last answer in the instruction generally contains the guiding information, we only construct the target word set and perform the perturbation operation for the last answer in the instruction for each instance.

is because the impact of single word substitution is subtle on the overall intention of whole instruction.

Forward Process of the Navigator. After introducing the forward process of the instruction attacker π_ϕ, we present the forward process of the navigator in this subsection. Specifically, the navigator follows an encoder-decoder architecture, where both the encoder and decoder are LSTMs [7]. The encoder contains a word embedding layer and a bi-directional LSTM, and its output is the language feature $\{\tilde{\mathbf{u}}_l\}_{l=1}^{L}$ of the instruction:

$$\begin{aligned}\mathbf{u}_l &= \text{Embedding}(w_l),\\ \tilde{\mathbf{u}}_1, \tilde{\mathbf{u}}_2, ..., \tilde{\mathbf{u}}_L &= \text{Bi}-\text{LSTM}(\mathbf{u}_1, \mathbf{u}_2, ...\mathbf{u}_L).\end{aligned} \tag{10}$$

Then, the decoder receives the attended visual feature $\mathbf{f}_t^v$ and language feature $\tilde{\mathbf{u}}$, and generates the visual-and-instruction aware hidden state $\tilde{\mathbf{h}}_t$:

$$\mathbf{h}_t = \text{LSTM}([\mathbf{f}_t^v; \mathbf{a}'_{t-1}], \tilde{\mathbf{h}}_{t-1}), \tag{11}$$

$$\alpha_{t,l}^w = \text{Softmax}(\tilde{\mathbf{u}}_l \mathbf{W}_u \mathbf{h}_t), \tag{12}$$

$$\mathbf{f}_t^w = \sum_l \alpha_{t,l}^w \tilde{\mathbf{u}}_l, \tag{13}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_{h'}[\mathbf{f}_t^w; \mathbf{h}_t]), \tag{14}$$

where $\mathbf{a}'_{t-1}$ is the action feature of the timestep $t-1$. $\mathbf{W}_u \in R^{D_w \times D_h}$ and $\mathbf{W}_{h'} \in R^{(D_w+D_h)\times D_h}$ are the learnable linear transformations. The attended visual feature $\mathbf{f}_t^v$ is calculated by:

$$\alpha_{t,i}^v = \text{Softmax}(\mathbf{f}_{t,i}^v \mathbf{W}_{v'} \tilde{\mathbf{h}}_{t-1}), \tag{15}$$

$$\mathbf{f}_t^v = \sum_i \alpha_{t,i}^v \mathbf{f}_{t,i}^v, \tag{16}$$

where $\mathbf{W}_{v'} \in R^{D_v \times D_h}$ is the learnable linear transformation. Then, the action prediction probability $\mathbf{p}_n(\mathbf{a}'_t)$ of the navigator is calculated by:

$$\mathbf{p}_n(\mathbf{a}'_t) = \text{Softmax}(\mathbf{c}'_{t,k} \mathbf{W}_a \tilde{\mathbf{h}}_t), \tag{17}$$

where $\mathbf{c}'_{t,k}$ denotes the candidate action features. $\mathbf{W}_a \in R^{D_v \times D_h}$ is the trainable linear transformation. The navigator takes the action a'_t according to the action prediction probability $\mathbf{p}_n(\mathbf{a}'_t)$.

The forward processes of the attacker and the navigator are shown in Figure 2. As illustrated in Figure 2, based on the attack score $\mathbf{p}_a(\mathbf{a}_t)$ which is calculated by the elementwise multiplication of word importance vector $\boldsymbol{\beta}_t$ and substitution impact matrix $\boldsymbol{\gamma}_t$, the perturbation operation is conducted on the original instruction $\tilde{\mathbf{u}}$ to generate perturbed instruction $\tilde{\mathbf{u}}'$. Then the decoder receives the attended visual feature $\mathbf{f}_t^v$ and the perturbed instruction $\tilde{\mathbf{u}}'$ to predict the next action a'_t. The updated hidden state $\tilde{\mathbf{h}}_t$ of the decoder and the target word feature $\mathbf{f}^w$ are used to calculate the prediction probability $\mathbf{p}_c(\mathbf{c})$ of the actual attacked word for the self-supervised auxiliary reasoning task.

Construction Details of Target Word Set and Candidate Word Set. In this part, we show the construction details of target word set and candidate substitution word set for both VLN and NDH tasks. Specifically, for each instruction, we first construct its target word set by conducting string match between it and the instruction vocabulary. The instruction vocabulary contains the words indicating visual objects or locations, which are collected from the given instruction vocabulary from the dataset. Then, the candidate substitution word set is constructed for each target word by selecting

TABLE 1: The comparison results with state-of-the-art methods on R2R dataset. Apart from NE, higher value indicates better results.

Method	Val Seen			Val Unseen			Test Unseen		
	NE(m) ↓	SR(%) ↑	SPL(%) ↑	NE(m) ↓	SR(%) ↑	SPL(%) ↑	NE(m) ↓	SR (%) ↑	SPL(%) ↑
seq-2-seq [1]	6.01	39	-	7.81	22	-	7.85	20	18
Speaker-Follower [6]	3.36	66	-	6.62	35	-	6.62	35	28
Regretful [54]	**3.23**	69	63	5.32	50	41	5.69	48	40
RCM [20]	3.53	67	-	6.09	43	-	6.12	43	38
PRESS [24]	4.39	58	55	5.28	49	45	5.59	49	45
EnvDrop [7]	3.99	62	59	5.22	52	48	**5.23**	51	47
Ours	3.52	**70**	**67**	**4.99**	**53**	**48**	5.53	**52**	**49**

TABLE 2: The comparison results with state-of-the-art methods on CVDN dataset. The Goal Progress (GP) (m) is reported following most existing works.

Method	Val Seen			Val Unseen			Test Unseen		
	Oracle	Navigator	Mixed	Oracle	Navigator	Mixed	Oracle	Navigator	Mixed
sequence-to-sequence [2]	4.48	5.67	5.92	1.23	1.98	2.10	1.25	2.11	2.35
CMN [22]	5.47	6.14	7.05	2.68	2.28	2.97	2.69	2.26	2.95
PREVALENT [23]	-	-	-	2.58	2.99	3.15	1.67	2.39	2.44
Ours	**5.60**	**7.58**	**8.06**	**3.27**	**4.00**	**4.18**	**2.77**	**2.95**	**3.26**

TABLE 3: The comparison of training time, data and device between PREVALENT [23] and our method on NDH.

Method	Time (min)			Data		Device	
	Pretrain	Other phases	Total	Pretrain	Other phases	Pretrain	Other phases
PREVALENT [23]	-	1661	-	6, 582, 000	4, 742	8 v100 GPUs	1 1080Ti GPU
Ours	143	328	471	4, 742	4, 742	1 1080Ti GPU	1 1080Ti GPU

the remained target words in the same instruction. The construction details of the target word set and candidate substitution word set for VLN and NDH tasks are shown in Figure 3 and Figure 4, respectively. Note that since the last answer in the dialog history plays the direct role of guiding navigation in the NDH task, we only construct the target word set and conduct the perturbation for the last answer in the NDH task, as shown in Figure 4.

D Experiment

In this section, we first introduce the datasets we use on NDH and VLN tasks, evaluation metrics, and implementation details in Sec. D.1. Then we provide the quantitative and qualitative results in Sec. D.2 and Sec. D.3, respectively.

D.1 Experimental Setup

D.1.1 Datasets

CVDN dataset [2] contains 2050 human-human navigation dialogs and over 7k trajectories in 83 MatterPort houses. Each trajectory is punctuated by several question-answer exchanges. Each dialog begins with an ambiguous instruction, and the subsequent dialog interaction between the navigator and oracle leads the navigator to find the target position. **R2R** dataset [1] includes 10,800 panoramic views and 7,189 trajectories. Each panoramic view has 36 images and each trajectory is paired with three natural language instructions. Both CVDN and R2R datasets are split into a training set, a seen validation set, an unseen validation set, and a test set.

D.1.2 Evaluation Metrics

The following four metrics [1] are used for evaluation on R2R dataset: 1) Trajectory Length (TL) measures the average length of navigation trajectories in meters, 2) Navigation Error (NE) is the distance between target viewpoint and agent stopping position, 3) Success Rate (SR) calculates the success rate of reaching the goal, 4) Success rate weighted by Path Length (SPL) makes the trade-off between SR and TL. Based on the metrics on R2R dataset, there are some new metrics used for evaluation on CVDN dataset [22]: 1) Goal Progress (GP) measures the average agent progress towards the goal location, 2) Oracle Success Rate (OSR) is the success rate if the agent can stop at the nearest point to the goal along its trajectory, 3) Oracle Path Success Rate (OPSR) means the success rate if the agent can stop at the closest point to the goal along the shortest path.

D.1.3 Implementation Details

The navigator architecture, training hyperparameters and the training strategy we use in both VLN and NDH tasks are the same to [7]. D_w, $D_{w'}$, D_v, D_p and D_h are set as 512, 512, 2052, 512 and 512, respectively. The positive/negative rewards of the final step and each non-stop step are set as 3/-3 and 1/-1, respectively. For both VLN and NDH, we split the training process for four steps: 1) pre-train the navigator using the original training set 2) pre-train the DR-Attacker on the pre-trained navigator and keep the parameters of the navigator fixed 3) adversarially train both the navigator and DR-Attacker by alternative iteration 4) finetune the navigator on the original training set. The

TABLE 4: The ablation study results on R2R dataset. NE (m), SR (%), SPL (%) results are reported. Apart from NE, higher value indicates better results.

Method	Val Seen			Val Unseen		
	NE (m) ↓	SR (%) ↑	SPL (%) ↑	NE (m) ↓	SR (%) ↑	SPL (%) ↑
Base Agent	4.37	58.7	56	5.43	48.0	45
DR-Attacker	4.55	57.3	55	5.62	46.6	43
Adversarial Training w auxiliary task	4.15	62.0	59	5.25	49.6	46
Finetune	3.52	70.2	67	4.99	53.2	48

TABLE 5: The ablation study results on CVDN dataset. The Goal Progress (GP) (m) is reported. The supervision setting is mixed supervision.

Settings	Val Seen			Val Unseen		
	Last A	Last QA	All	Last A	Last QA	All
Base Agent	7.55	7.15	7.35	3.88	3.95	3.72
DR-Attacker	5.59	5.88	5.85	1.98	2.51	2.29
Adversarial Training w auxiliary task	6.80	6.96	7.23	3.90	3.80	3.93
Finetune	7.66	7.61	8.06	4.20	4.19	4.18

TABLE 6: The ablation study results on CVDN dataset. The Goal Progress (GP) (m) is reported. The dialog history setting is last answer.

Settings	Val Seen			Val Unseen		
	Oracle	Navigator	Mixed	Oracle	Navigator	Mixed
Base Agent	5.44	6.92	7.55	3.28	4.06	3.88
DR-Attacker	4.00	5.07	5.30	1.50	2.38	1.95
Adversarial Training	5.48	7.13	7.61	3.37	4.16	4.08
Adversarial Training w auxiliary task	5.52	7.49	7.66	3.48	4.21	4.20

TABLE 7: The comparison results of different adversarial attacking mechanisms in attacking and promoting the navigation performance on CVDN dataset. The Goal Progress (GP) (m) is reported. The dialog history setting is last answer. Ora., Nav., and Mix. represent the supervision is Oracle, Navigator and Mixed, respectively.

Method	Val Seen			Val Unseen		
	Ora.	Nav.	Mix.	Ora.	Nav.	Mix.
Direct Attack						
Static	4.09	5.32	5.48	1.71	2.63	2.23
Random	4.20	5.71	5.66	1.64	2.54	2.15
Heuristics	4.07	**4.99**	5.35	1.52	2.46	**1.89**
PWWS [12]	4.04	5.51	5.44	1.64	2.59	2.10
DR-Attacker	**4.00**	5.07	**5.30**	**1.50**	**2.38**	1.95
Adversarial Training						
Static	5.42	6.57	7.09	3.25	3.93	3.97
Random	4.89	6.52	7.02	3.15	3.93	3.88
Heuristics	**5.60**	6.91	6.99	3.04	3.81	3.67
PWWS [12]	5.23	6.57	6.87	3.20	3.59	3.09
DR-Attacker	5.52	**7.49**	**7.66**	**3.48**	**4.21**	**4.20**

training iterations of four steps for VLN are 40K, 10K, 40K, 200K and the training iterations of four steps for NDH are 5K, 1K, 3K, 3K. For the adversarial training, the alternation is conducted after 3K and 1K iterations for VLN and NDH, respectively. Following [6], we also use the data augmentation of instruction to improve the navigation performance. For improving the learning efficiency, we also introduce imitation learning supervision [7] when training the navigator in the adversarial training stage.

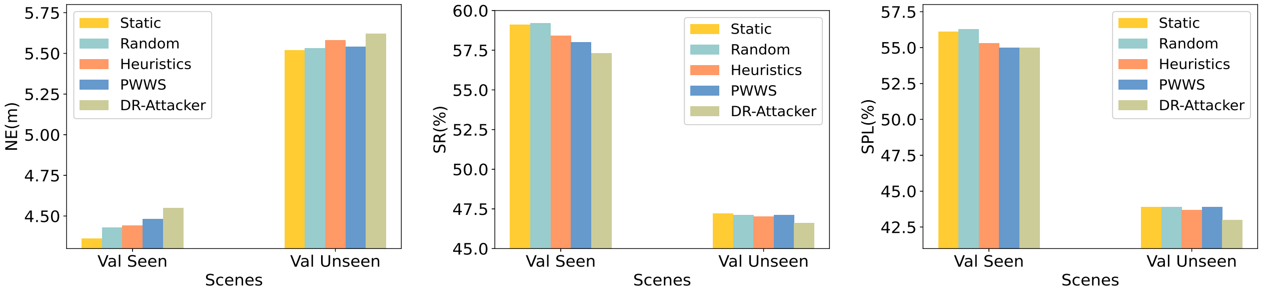
D.2 Quantitative Results

D.2.1 Comparison with the State-of-the-art Methods

The quantitative comparison results with state-of-the-art methods on VLN and NDH are given in Table 1 and Table 2, respectively. In Table 1, we report three most important metrics in the VLN setting, i.e., Navigation Error (NE), Success Rate (SR) and Success rate weighted by Path Length (SPL). In Table 2, we report the Goal Progress (GP) metric under the whole dialog history setting following most existing works on VDN [2], [22], [23].

Table 1 indicates that our proposed method outperforms other competitors in most metrics. Comparing with the baseline EnvDrop [7], the improvements for the SR and SPL of our method are significant in both seen and unseen settings. Table 2 shows that our method outperforms the state-of-the-art methods by a significant margin on NDH in both seen and unseen environments. We further compare the training time, data and device between the state-of-the-art method PREVALENT [23] and our method on NDH. Since only the implementation of finetuning phase[1] is available for PREVALENT [23], we only record the reimplemented finetuning time of PREVALENT [23] for comparison. Other values for the pretraining phase of PREVALENT [23] are the reported values in their paper. The results are given in Table 3. From Table 3 we can find that compared with PREVALENT [23], our proposed method need significantly

[1] https://github.com/weituo12321/PREVALENT_R2R



(a) NE(m) for different scenes

(b) SR(%) for different scenes

(c) SPL(%) for different scenes

Figure 5: The comparison results of different types of adversarial attacking mechanisms on VLN. NE (m), SR (%) and SPL (%) are reported for both Val Seen and Val Unseen scenes. Apart from NE, lower value indicates better results.

less training time, data, and computation resource while can achieve better results, showing the good flexibility of our method. Both the results on VLN and NDH show the effectiveness of the proposed method in improving the robustness of the navigation agent.

D.2.2 Ablation Study

In this section, we conduct ablation study to validate the effectiveness of the proposed adversarial attacking paradigm, adversarial training strategy and the auxiliary self-supervised reasoning task. Specifically, the effects of four-stage training for VLN and NDH tasks are presented in Table 4 and Table 5. The effectiveness of the auxiliary self-supervised reasoning task is given in Table 6. For VLN, "Base Agent" means pre-training navigators on the datasets composing of original instructions and augmented instructions for 40K iterations. "Finetune" means finetuning the adversarial trained agents on the same dataset as that used in the pretraining stage. For VDN, "Base Agent" means using the same training strategy like [7] to pre-train the navigators on the original dataset for 5k iterations. "Finetune" means finetuning the adversarial trained agents on the original dataset. "DR-Attacker" represents the navigation results when receiving perturbed instructions. "Last A", "Last QA" and "All" represent three kinds of different dialog history settings, i.e., the instruction is last answer, last question-answer pair or the whole dialog history [2].

From Table 4 and Table 5 we can find that our proposed four-stage training strategy can effectively contribute to enhancing the robustness and the navigation performance of the agent on both VLN and NDH tasks. Specifically, by introducing adversarial perturbations on the instructions, the navigation performance of the agent shows significant drop, demonstrating the effectiveness of the proposed adversarial attacking mechanism. Then, after adversarial training with the proposed auxiliary self-supervised reasoning task followed by finetuning on the original dataset, the robustness and the navigation performance can be effectively improved. Moreover, from Table 6 we can observe that by introducing our proposed self-supervised auxiliary reasoning task in the adversarial training stage, the navigation performance can be effectively enhanced, demonstrating that improving the cross-modality understanding ability of the agent is crucial for successful navigation.

D.2.3 Different Types of Attacking Mechanisms

In this subsection, we compare different types of attacking mechanisms to validate the effectiveness of the proposed DR-Attacker and in attacking and promoting the navigation performance through adversarial training. Specifically, four adversarial attacking methods or variants are chosen for the comparison: 1) "Static" means that the perturbation at each timestep is invariant, i.e., at each timestep, the same target word is substituted with the same candidate word. For selecting the target word and candidate word, we use the pretrained DR-Attacker to conduct the word prediction at the first navigation timestep. 2) "Random" represents randomly selecting the target word and the candidate substitution word at each timestep. 3)"Heuristics" means the instruction word that receives the highest textual attention weights from the navigator at each timestep is destroyed. 4) PWWS [12] is an adversarial attack method in NLP which is similar to our proposed adversarial attack in some implementation procedures. It also obtains an attack score by calculating word importance and substitution impact according to the change of classification probability. Since there is no direct classification-based objective for the instruction in both VLN and NDH tasks, we choose the action prediction probability for an alternative. Specifically, at each timestep, the attacked word which can cause the maximum change of the original action prediction probability is destroyed. Therefore, "Random", "Heursitics" and PWWS are all dynamic adversarial attacks.

The comparison results of attacking effects on VLN and NDH tasks are given in Figure 5 and Table 7, respectively. And the adversarial training results using different attacking mechanisms on NDH are given in Table 7. From Figure 5 and Table 7 we can find that compared with either static or dynamic attacking mechanisms, our proposed DR-Attacker can achieve the best attack results in most metrics on both VLN and NDH tasks, demonstrating the importance of dynamically attacking key information in the navigation task and the effectiveness of our proposed RL-based optimization method for the proposed adversarial attack. Moreover, from the adversarial training results in Table 7 we can find the superiority of DR-Attacker in promoting the navigation performance compared with other attacking methods, demonstrating that jointly optimizing the navigator and the attacker is more beneficial for the improvement of the navigation performance. Both the attacking and adversarial training results on VLN and NDH tasks show the effective-

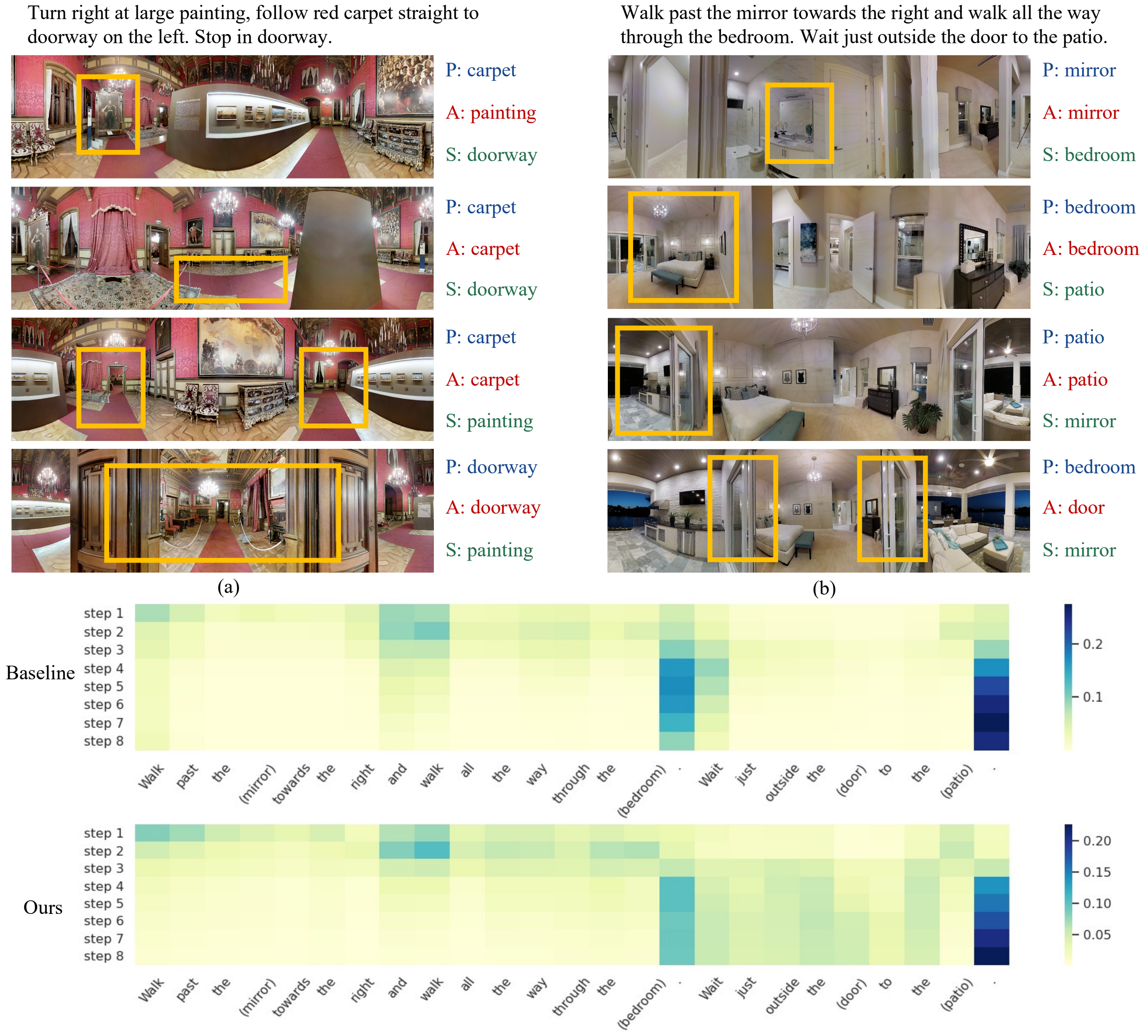
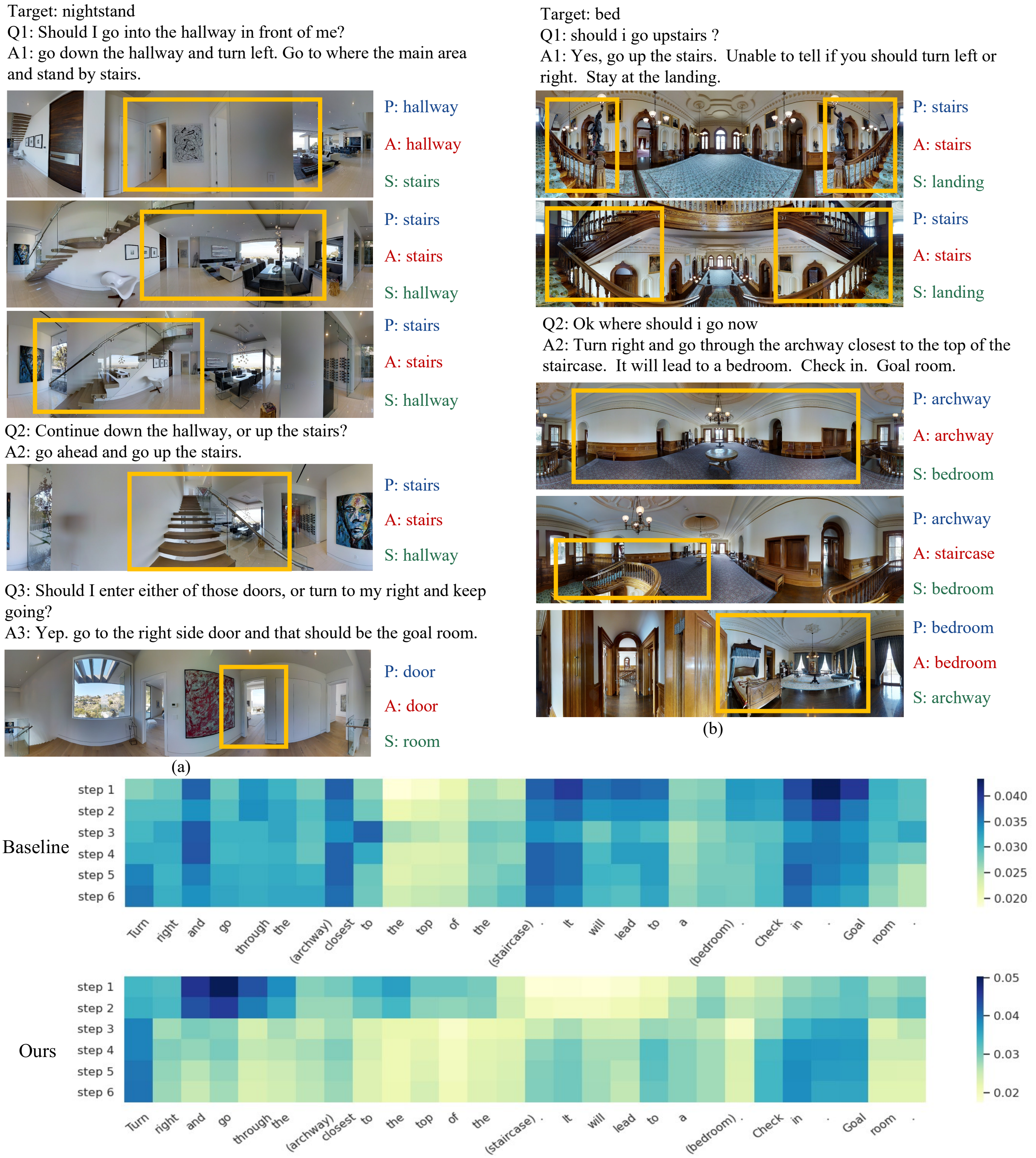


Fig. 6: The visualization examples of perturbed instructions, panoramic views, and language attention weights (instance (b)) during trajectories on VLN. The words in red, blue and green color represent the actual attacked word by DR-Attacker (A), the predicted attacked word by the navigator (P), and the substitution word (S), respectively. Yellow bounding box denotes the visual object or location at the current scene. "Baseline" and "Ours" represent the navigators trained without and with perturbed instructions, respectively. Words in the bracket represent the actual attacked word by the DR-Attacker. Best viewed in color.

ness of the proposed adversarial attacking mechanism and adversarial training paradigm.

D.3 Qualitative Results

In this subsection, we show the visualization examples of perturbed instructions, panoramic views and language attention weights during trajectories on VLN and NDH tasks. The results are given in Figure 6 and Figure 7, respectively. From Figure 6 and Figure 7 we can find that the proposed DR-Attacker can successfully locate the word which appears in the scene at different timesteps and substitute it with the word that doesn't exist in the current scene. Moreover, the navigator can make correct predictions of the actual attacked words by DR-Attacker, showing its good understanding of the multi-modality observations. The first subfigure in Figure 6 (a), the fourth subfigure in Figure 6 (b) and the second subfigure in Figure 7 (a) show the failure cases. From the failure cases, we can find that when there are multiple objects referred in the instruction simultaneously existing in the current scene, e.g., both the "bedroom" and "door" exist in the fourth subfigure in Figure 6 (b), the navigator or the DR-Attacker may be confused. From the language attention weights of the navigators trained with perturbed instructions ("Ours"), we can find that although



Target: nightstand
Q1: Should I go into the hallway in front of me?
A1: go down the hallway and turn left. Go to where the main area and stand by stairs.

Q2: Continue down the hallway, or up the stairs?
A2: go ahead and go up the stairs.

Q3: Should I enter either of those doors, or turn to my right and keep going?
A3: Yep. go to the right side door and that should be the goal room.

Target: bed
Q1: should i go upstairs ?
A1: Yes, go up the stairs. Unable to tell if you should turn left or right. Stay at the landing.

Q2: Ok where should i go now
A2: Turn right and go through the archway closest to the top of the staircase. It will lead to a bedroom. Check in. Goal room.

Fig. 7: The visualization examples of perturbed instructions, panoramic views, and language attention weights (A2 in instance (b)) during trajectories on NDH. The words in red, blue and green color represent the actual attacked word by DR-Attacker (A), the predicted attacked word by the navigator (P), and the substitution word (S), respectively. Yellow bounding box denotes the visual object or location at the current scene. "Baseline" and "Ours" represent the navigators trained without and with perturbed instructions, respectively. Words in the bracket represent the actual attacked word by the DR-Attacker. Best viewed in color.

the target word is attacked, the navigator can attend to the context near the attacked word to capture the language intention. Moreover, with the process of the navigation trajectory, it can successfully capture important instruction information in different phases. In contrast, the navigator trained without perturbed instructions ("Baseline") generates a confused language attention weights by the introduced perturbations during navigation. These visualization analyses show that emphasizing useful instruction information can contribute to successful navigation. Moreover, our proposed adversarial attacking and adversarial training mechanisms can effectively improve the robustness of the navigation agent.

E Conclusion

In this work, we propose Dynamic Reinforced Instruction Attacker (DR-Attacker) for the natural language grounded visual navigation tasks. By formulating the perturbation generation using the RL framework, DR-Attacker can be optimized iteratively to capture the crucial parts in instructions and generate meaningful adversarial samples. Through adversarial training using perturbed instructions, the robustness of the navigator can be effectively enhanced with an auxiliary self-supervised reasoning task. Experiments on both VLN and NDH tasks show the effectiveness of the proposed method.

In the future, we plan to improve the training strategy of the proposed instruction attacker and exploit to design more effective attacks on the navigation instruction. Moreover, we would like to develop multi-modality adversarial attacks for the embodied navigation task to further verify and improve the robustness of the navigator.

References

[1] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sunderhauf, I. Reid, S. Gould, and A. van den Hengel, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3674–3683.

[2] J. Thomason, M. Murray, M. Cakmak, and L. Zettlemoyer, "Vision-and-dialog navigation," *Conference on Robot Learning (CoRL)*, pp. 394–406, 2019.

[3] Y. Qi, Q. Wu, P. Anderson, X. Wang, W. Y. Wang, C. Shen, and A. van den Hengel, "Reverie: Remote embodied visual referring expression in real indoor environments," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 9982–9991.

[4] K. Nguyen and H. Daumé, "Help, anna! vision-based navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning," in *2019 Conference on Empirical Methods in Natural Language Processing*, 2019, pp. 684–695.

[5] H. Chen, A. Suhr, D. Misra, N. Snavely, and Y. Artzi, "Touchdown: Natural language navigation and spatial reasoning in visual street environments," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 12 538–12 547.

[6] D. Fried, R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L.-P. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell, "Speaker-follower models for vision-and-language navigation," in *NIPS 2018: The 32nd Annual Conference on Neural Information Processing Systems*, 2018, pp. 3314–3325.

[7] H. Tan, L. Yu, and M. Bansal, "Learning to navigate unseen environments: Back translation with environmental dropout," in *NAACL-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2019, pp. 2610–2621.

[8] T.-J. Fu, X. E. Wang, M. F. Peterson, S. T. Grafton, M. P. Eckstein, and W. Y. Wang, "Counterfactual vision-and-language navigation via adversarial path sampler." in *European Conference on Computer Vision*, 2020, pp. 71–86.

[9] V. Araujo, A. Carvallo, C. Aspillaga, and D. Parra, "On adversarial examples for biomedical nlp tasks," *arXiv preprint arXiv:2004.11157*, 2020.

[10] L. Li, R. Ma, Q. Guo, X. Xue, and X. Qiu, "Bert-attack: Adversarial attack against bert using bert," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 6193–6202.

[11] J. Ebrahimi, D. Lowd, and D. Dou, "On adversarial examples for character-level neural machine translation," in *COLING 2018: 27th International Conference on Computational Linguistics*, 2018, pp. 653–663.

[12] S. Ren, Y. Deng, K. He, and W. Che, "Generating natural language adversarial examples through probability weighted word saliency," in *ACL 2019 : The 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 1085–1097.

[13] Y. Zang, F. Qi, C. Yang, Z. Liu, M. Zhang, Q. Liu, and M. Sun, "Word-level textual adversarial attacking as combinatorial optimization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 6066–6080.

[14] X. E. Wang, V. Jain, E. Ie, W. Y. Wang, Z. Kozareva, and S. Ravi, "Environment-agnostic multitask learning for natural language grounded navigation," in *ECCV (24)*, 2020, pp. 413–430.

[15] K. Nguyen, D. Dey, C. Brockett, and B. Dolan, "Vision-based navigation with language-based assistance via imitation learning with indirect intervention," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 12 527–12 537.

[16] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh, "Vqa: Visual question answering," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2425–2433.

[17] H. de Vries, F. Strub, S. Chandar, O. Pietquin, H. Larochelle, and A. Courville, "Guesswhat?! visual object discovery through multi-modal dialogue," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 4466–4475. [Online]. Available: https://academic.microsoft.com/paper/2558809543

[18] A. Das, S. Kottur, K. Gupta, A. Singh, D. Yadav, J. M. F. Moura, D. Parikh, and D. Batra, "Visual dialog," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. [Online]. Available: https://academic.microsoft.com/paper/2768661419

[19] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, "Image captioning with semantic attention," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4651–4659.

[20] X. Wang, Q. Huang, A. Celikyilmaz, J. Gao, D. Shen, Y.-F. Wang, W. Y. Wang, and L. Zhang, "Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 6629–6638.

[21] C.-Y. Ma, jiasen lu, Z. Wu, G. AlRegib, Z. Kira, richard socher, and C. Xiong, "Self-monitoring navigation agent via auxiliary progress estimation," in *ICLR 2019 : 7th International Conference on Learning Representations*, 2019.

[22] Y. Zhu, F. Zhu, Z. Zhan, B. Lin, J. Jiao, X. Chang, and X. Liang, "Vision-dialog navigation by exploring cross-modal memory," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 10 730–10 739.

[23] W. Hao, C. Li, X. Li, L. Carin, and J. Gao, "Towards learning a generic agent for vision-and-language navigation via pre-training," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 13 137–13 146.

[24] X. Li, C. Li, Q. Xia, Y. Bisk, A. Çelikyilmaz, J. Gao, N. A. Smith, and Y. Choi, "Robust navigation with language pretraining and stochastic sampling." in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 1494–1499.

[25] T. Cemgil, S. Ghaisas, K. Dvijotham, and P. Kohli, "Adversarially robust representations with smooth encoders," in *ICLR 2020 : Eighth International Conference on Learning Representations*, 2020.

[26] H. Zhang and J. Wang, "Defense against adversarial attacks using feature scattering-based adversarial training," in *NeurIPS 2019 : Thirty-third Conference on Neural Information Processing Systems*, 2019, pp. 1831–1841.

[27] H. Salman, J. Li, I. Razenshteyn, P. Zhang, H. Zhang, S. Bubeck, and G. Yang, "Provably robust deep learning via adversarially trained smoothed classifiers," in *NeurIPS 2019 : Thirty-third Conference on Neural Information Processing Systems*, 2019, pp. 11 292–11 303.

[28] J. Feng, Q.-Z. Cai, and Z.-H. Zhou, "Learning to confuse: Generating training time adversarial data with auto-encoder," in *NeurIPS 2019 : Thirty-third Conference on Neural Information Processing Systems*, 2019, pp. 11 994–12 004.

[29] K. R. Mopuri, A. Ganeshan, and R. V. Babu, "Generalizable data-free objective for crafting universal adversarial perturbations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 10, pp. 2452–2465, 2019.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR 2015 : International Conference on Learning Representations 2015*, 2015.

[32] D. Wang, C. Gong, and Q. Liu, "Improving neural language modeling via adversarial training," in *ICML 2019 : Thirty-sixth International Conference on Machine Learning*, 2019, pp. 6555–6565.

[33] C. Zhu, Y. Cheng, Z. Gan, S. Sun, T. Goldstein, and J. Liu, "Freelb: Enhanced adversarial training for natural language understanding," in *International Conference on Learning Representations*, 2019.

[34] Y. Cheng, L. Jiang, and W. Macherey, "Robust neural machine translation with doubly adversarial inputs," in *ACL 2019 : The 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 4324–4333.

[35] E. Jones, R. Jia, A. Raghunathan, and P. Liang, "Robust encodings: A framework for combating adversarial typos." in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 2752–2765.

[36] J. Ebrahimi, A. Rao, D. Lowd, and D. Dou, "Hotflip: White-box adversarial examples for text classification," in *ACL 2018: 56th Annual Meeting of the Association for Computational Linguistics*, vol. 2, 2018, pp. 31–36.

[37] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 670–680.

[38] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *NAACL-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2019, pp. 4171–4186.

[39] B. Wang, S. Wang, Y. Cheng, Z. Gan, R. Jia, B. Li, and J. Liu, "Infobert: Improving robustness of language models from an information theoretic perspective," in *ICLR 2021: The Ninth International Conference on Learning Representations*, 2021.

[40] R. Jia, A. Raghunathan, K. Göksel, and P. Liang, "Certified robustness to adversarial word substitutions," in *2019 Conference on Empirical Methods in Natural Language Processing*, 2019, pp. 4127–4140.

[41] S. Eger, G. G. Sahin, A. Rücklé, J.-U. Lee, C. Schulz, M. Mesgar, K. Swarnkar, E. Simpson, and I. Gurevych, "Text processing like humans do: Visually attacking and shielding nlp systems," in *NAACL-HLT 2019: Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2019, pp. 1634–1647.

[42] X. Liu, H. Cheng, P. He, W. Chen, Y. Wang, H. Poon, and J. Gao, "Adversarial training for large neural language models," *arXiv preprint arXiv:2004.08994*, 2020.

[43] F. Yin, Q. Long, T. Meng, and K.-W. Chang, "On the robustness of language encoders against grammatical errors," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 3386–3403.

[44] A. Liu, T. Huang, X. Liu, Y. Xu, Y. Ma, X. Chen, S. J. Maybank, and D. Tao, "Spatiotemporal attacks for embodied agents." in *ECCV (17)*, 2020, pp. 122–138.

[45] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, "Embodied question answering," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018, pp. 1–10.

[46] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "Autoaugment: Learning augmentation strategies from data," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 113–123.

[47] D. Ho, E. Liang, X. Chen, I. Stoica, and P. Abbeel, "Population based augmentation: Efficient learning of augmentation policy schedules," in *International Conference on Machine Learning*, 2019, pp. 2731–2741.

[48] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim, "Fast autoaugment," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 6665–6675.

[49] R. Hataya, J. Zdenek, K. Yoshizoe, and H. Nakayama, "Faster autoaugment: Learning augmentation strategies using backpropagation." in *ECCV (25)*, 2020, pp. 1–16.

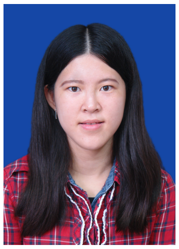
[50] E. D. Cubuk, B. Zoph, J. Shlens, and Q. Le, "Randaugment: Practical automated data augmentation with a reduced search space," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 18 613–18 624.

[51] Y. Wu, Y. Wu, G. Gkioxari, and Y. Tian, "Building generalizable agents with a realistic and rich 3d environment," in *ICLR 2018 : International Conference on Learning Representations 2018*, 2018.

[52] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Harley, T. P. Lillicrap, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *ICML'16 Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, 2016, pp. 1928–1937.

[53] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust adversarial reinforcement learning," in *ICML'17 Proceedings of the 34th International Conference on Machine Learning - Volume 70*, 2017, pp. 2817–2826.

[54] C.-Y. Ma, Z. Wu, G. AlRegib, C. Xiong, and Z. Kira, "The regretful agent: Heuristic-aided navigation through progress estimation," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 6732–6740.



Bingqian Lin received the B.E. and the M.E. degree in Computer Science from University of Electronic Science and Technology of China and Xiamen University, in 2016 and 2019, respectively. She is currently working toward the D.Eng in the school of intelligent systems engineering of Sun Yat-sen University. Her research interests include multi-view clustering, image processing and vision-and-language understanding.

Yi Zhu received the B.S. degree in software engineering from Sun Yat-sen University, Guangzhou, China, in 2013. Since 2015, she has been a Ph.D student in computer science with the School of Electronic, Electrical, and Communication Engineering, University of Chinese Academy of Sciences, Beijing, China. Her current research interests include object recognition, scene understanding, weakly supervised learning and visual reasoning.

Yanxin Long is a first-year master in the School of Intelligent Systems Engineering, Sun Yat-Sen University. He works at the Human Cyber Physical Intelligence Integration Lab under the supervision of Prof. Xiaodan Liang. Before that, He received my Bachelor Degree from the Comunication College, Xidian University in 2020. His research interests include reinforcement learning and vision-and-language understanding.

Xiaodan Liang is currently an Associate Professor at Sun Yat-sen University. She was a postdoc researcher in the machine learning department at Carnegie Mellon University, working with Prof. Eric Xing, from 2016 to 2018. She received her PhD degree from Sun Yat-sen University in 2016, advised by Liang Lin. She has published several cutting-edge projects on human-related analysis, including human parsing, pedestrian detection and instance segmentation, 2D/3D human pose estimation and activity recognition.

Qixiang Ye received the B.S. and M.S. degrees in mechanical and electrical engineering from the Harbin Institute of Technology, China, in 1999 and 2001, respectively, and the Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, in 2006. He has been a Professor with the University of Chinese Academy of Sciences since 2009, and was a Visiting Assistant Professor with the Institute of Advanced Computer Studies, University of Maryland, College Park, in 2013. He has authored over 50 papers in refereed conferences and journals, and received the Sony Outstanding Paper Award. His current research interests include image processing, visual object detection and machine learning. He pioneered the Kernel SVM-based pyrolysis output prediction software which was put into practical application by SINOPEC in 2012. He developed two kinds of piecewise linear SVM methods which were successfully applied into visual object detection.

Liang Lin is CEO of DMAI Great China and a full professor of Computer Science in Sun Yat-sen University. He served as the Executive Director of the SenseTime Group from 2016 to 2018, leading the R&D teams in developing cutting-edge, deliverable solutions in computer vision, data analysis and mining, and intelligent robotic systems. He has authored or co-authored more than 200 papers in leading academic journals and conferences (e.g., TPAMI/IJCV, CVPR/ICCV/NIPS/ICML/AAAI). He is an associate editor of IEEE Trans, Human-Machine Systems and IET Computer Vision, and he served as the area/session chair for numerous conferences, such as CVPR, ICME, ICCV, ICMR. He was the recipient of Annual Best Paper Award by Pattern Recognition (Elsevier) in 2018, Dimond Award for best paper in IEEE ICME in 2017, ACM NPAR Best Paper Runners-Up Award in 2010, Google Faculty Award in 2012, award for the best student paper in IEEE ICME in 2014, and Hong Kong Scholars Award in 2014. He is a Fellow of IET.