DIAGNOSING THE ENVIRONMENT BIAS IN VISION-AND-LANGUAGE NAVIGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-and-Language Navigation (VLN) requires an agent to follow natural-language instructions, explore the given environments, and reach the desired target locations. These step-by-step navigational instructions are extremely useful in navigating new environments which the agent does not know about previously. Most recent works that study VLN observe a significant performance drop when tested on unseen environments (i.e., environments not used in training), indicating that the neural agent models are highly biased towards training environments. Although this issue is considered as one of major challenges in VLN research, it is still under-studied and needs a clearer explanation. In this work, we design novel diagnosis experiments via environment re-splitting and feature replacement, looking into possible reasons of this environment bias. We observe that neither the language nor the underlying navigational graph, but the low-level visual appearance conveyed by ResNet features directly affects the agent model and contributes to this environment bias in results. According to this observation, we explore several kinds of semantic representations which contain less low-level visual information, hence the agent learned with these features could be better generalized to unseen testing environments. Without modifying the baseline agent model and its training method, our explored semantic features significantly decrease the performance gap between seen and unseen on multiple datasets (i.e., 8.6% to 0.2% on R2R, 23.9% to 0.1% on R4R, and 3.74 to 0.17 on CVDN) and achieve competitive unseen results to previous state-of-the-art models.

1 INTRODUCTION

Vision-and-Language Navigation (VLN) tests an agent's ability to follow complex natural language instructions as well as explore the given environments, so as to be able to reach the desired target locations. As shown in Fig. 1, the agent is put in an environment and given a detailed step-by-step navigational instruction. With these inputs, the agent needs to navigate the environment and find the correct path to the target location. In this work, we focus on the instruction-guided navigation (MacMahon et al., 2006; Anderson et al., 2018b; Misra et al., 2018; Blukis et al., 2018; Chen et al., 2019c) where detailed step-by-step navigational instructions are used (e.g., 'Go outside the dinning room and turn left ...'), in contrast to the target-oriented navigation (Gordon et al., 2018; Das et al., 2018; Mirowski et al., 2018; Yu et al., 2019) where only the target is referred (e.g., 'Go to the kitchen' or 'Tell me the color of the bedroom'). Although these step-by-step instructions are over-detailed when navigating local areas (e.g., your home), they are actively used in unseen environments (e.g., your friend's house, a new city) where the desired target is usually unknown to navigational agents. For this purpose, testing on unseen environments which are not used during agent-training is important and widely accepted by instruction-guided navigation datasets.

Recent works propose different methods to improve generalizability of agents on these unseen testing environments; and most of the existing works (Anderson et al., 2018b; Wang et al., 2018b; Fried et al., 2018; Wang et al., 2019b; Ma et al., 2019a;b; Tan et al., 2019; Huang et al., 2019; Hu et al., 2019) observe a significant performance drop from seen environments (i.e., the environments used in training) to unseen environments (i.e., the environments not used in training), which indicates a strong bias in the model towards the training environments. While this performance gap is emphasized as one of major challenges in current VLN research, the issue is still left unresolved and waits for an explicit explanation. Thus, in this paper, we aim to answer three questions to this environment

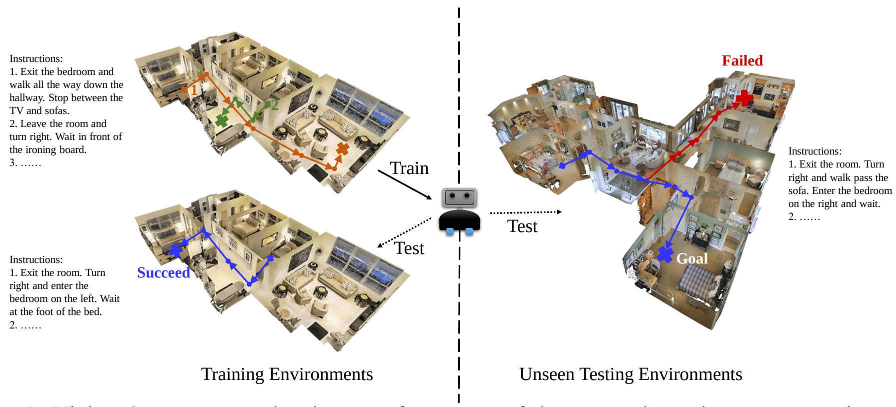


Figure 1: Vision-language-navigation: performance of the agent drops in unseen environments.

bias: 1. **Where** (i.e., in which component) is the bias located? 2. **Why** does this bias exist? 3. **How** to eliminate this bias?

To locate **where** the bias is, we start with showing that natural-language navigational instructions and underlying navigational graphs are not direct reasons for this performance gap. We then investigate the effect of environments on the agent's performance. In order to conduct a detailed analysis, we re-split the environment and categorize the validation data into three sets based on their visibility to the training set: *path-seen* data intersecting with the training paths, *path-unseen* data using the training environments but away from the training paths, and *env-unseen* data using unseen environments (environments not used in training). By showing that the results gradually decrease from path-seen data to env-unseen data, we characterize the environment bias at three levels: path level, region level, and environment level.

These three levels of environment biases indicate strong 'spatial localities' in the tasks of VLN, which are intuitively reasonable because environments and regions (e.g., houses and cities) usually have their own styles when built or decorated. We next want to analyze the detailed reason **why** this locality would further lead to a gap in seen versus unseen results. Our hypothesis links this reason to the *low-level information* carried by the ResNet features (He et al., 2016). To keep minimal low-level visual information and promote more high-level semantic information, we replace the ResNet features with the 1000 ImageNet classification probabilities. Although the semantic information encoded by this features is not accurate because of the shifted domain of images and labels, the same model with ImageNet-Labels features performs credibly well on various VLN datasets (i.e., Room-to-Room, R4R, and CVDN[1]). And most importantly, these noisy semantic features effectively eliminates the performance gap between seen and unseen environments, which suggests that the environment bias is attributed to the ResNet features as our hypothesis.

Following the practice in using ImageNet labels as semantic features, we further provide discussion on **how** the environment bias could be eliminated. For this, we employ advanced high-level semantic features which are more rational for the VLN domain. We explore three kinds of semantic features: (1) areas of detected object labels (Ren et al., 2015); (2) ground truth semantic views (Chang et al., 2017); and (3) learned semantic view features. We show that all of these semantic features significantly reduce the environment bias in multiple datasets and also achieve strong results on testing unseen environments. We hope this work encourages more investigation and research into improving the generalization of vision-language models to unseen real-world scenarios.

2 Related Work

Vision-and-Language Navigation: Vision-and-language navigation is an emerging task in the vision-and-language area. A lot of datasets have been proposed in recent years, such as Room-to-Room (Anderson et al., 2018b), Room-for-Room (Jain et al., 2019), TouchDown (Chen et al., 2019c), CVDN (Thomason et al., 2019b), RERERE (Qi et al., 2019), and EQA (Das et al., 2018). Recent works (Thomason et al., 2019a; Wang et al., 2018b; Fried et al., 2018; Wang et al., 2019b;

[1]We did not test these semantic features on touchdown (Chen et al., 2019c) since the images are not released.

Ma et al., 2019a;b; Tan et al., 2019; Thomason et al., 2019a; Hu et al., 2019; Ke et al., 2019; Anderson et al., 2019) focusing on improving the performance of navigation models, especially in unseen testing environments, have helped to increase the navigational success rate.

Domain Adaptation: The general setup of domain adaption contains two sets of data samples $\{x_i\}_{x_i \in X}$ and $\{y_i\}_{y_i \in Y}$ from two domains X and Y. Based on these samples, we could learn domain invariant feature with adversarial training (Goodfellow et al., 2014; Zhu et al., 2017; Long et al., 2018; Wang et al., 2019a; Hosseini-Asl et al., 2019; Zhang et al., 2019; Gong et al., 2019; Chen et al., 2019b) or learn a transfer function $f : X \rightarrow Y$ (Wang et al., 2018a; Chen et al., 2019a; Rozantsev et al., 2018). However, samples from the target domain may not be available (e.g., the testing environments in navigation should not be used in training) in real applications. Thus, we try to give an interpretable explanation to why performance varies in different domains and design a robust feature for it without deliberately considering the target domain. Two methods in VLN, RCM (Wang et al., 2019b) and EnvDrop (Tan et al., 2019), explore the possibility of domain adaptation. Both RCM and EnvDrop take the testing environments in training while RCM also uses testing instructions.

Domain Generalization: In domain generalization (Blanchard et al., 2011), the goal is to predict the labels in the previous unseen domain. Similar to the test setting of VLN tasks, the testing data is unrevealed in training. Works have been proposed to learn the common features of the training domain (Muandet et al., 2013; Blanchard et al., 2017; Li et al., 2017; 2018; Carlucci et al., 2019; Deshmukh et al., 2019). In this paper, we focus on the domain generalization problem in VLN task, and try to find the reasons of the failures.

3 VISION-AND-LANGUAGE NAVIGATION AND ITS ENVIRONMENT BIAS

We first introduce the task of vision-and-language navigation (VLN) and briefly discuss the neural agent models used in our work. We next survey previous works on multiple indoor navigation datasets to show that the environment bias are widely observed in current VLN research. Lastly, we claim that this bias also exists in the outdoor navigation tasks, if we test the agent on unseen areas.

3.1 VISION-AND-LANGUAGE NAVIGATION

Tasks: As shown in Fig. 1, the goal of the VLN task is to train an agent to navigate a certain type of environments $\{\mathbf{E}\}$ (e.g., indoor or outdoor environments) given the instruction $\mathbf{I}$. Each environment $\mathbf{E}$ is an independent space, such as a room or a house, and consists of a set of viewpoints. Each viewpoint is represented as a panoramic image and can be decomposed into separate views $\{o\}$ as inputs to the neural agent models. The viewpoints and their connectivity form the navigational graph. In practice, after being placed at a particular viewpoint and given the instruction in the beginning, at each time step, the agent can observe the panoramic image of the viewpoint where it is located, and choose to move along an edge of the graph to the next node (i.e., viewpoint) or stop. This navigational process produces a path (i.e., a list of viewpoints), and the performance of the agent is evaluated by whether it reaches the target location that the instruction indicates in the end.

Neural Agent Models: Most instruction-guided navigational agents are built based on attentive encoder-decoder models (Bahdanau et al., 2015). The encoder reads the instructions while the decoder outputs actions based on the encoded instructions and perceived environments. Since the main purpose of this work is to understand the environment bias in vision-and-language navigation, we use a minimal representative neural agent model which achieves comparable results to previous works. Specifically, we adopt the panoramic-view neural agent model in Fried et al. (2018) ('Follower') with modifications from Tan et al. (2019) as our baseline model. We also exclude advanced training techniques (i.e., reinforcement learning and data augmentation) and only train the agent with imitation learning in all our experiments for the same purpose. More details in original paper.

3.2 ENVIRONMENT BIAS IN INDOOR NAVIGATION

To evaluate the generalizability of agent models, indoor vision-and-language navigation datasets (e.g., those collected from Matterport3D (Chang et al., 2017)) use disjoint sets of environments in training and testing. And most of datasets provide two validation splits to verify the agent's per-

Table 1: Results showing the performance gap between seen ('Val Seen') and unseen ('Val Unseen') environments. Room-to-Room, Room-for-Room, and Touchdown are evaluated with 'Success Rate', CVDN is evaluated with 'Goal Progress'.

Task	**Method**	**Result**		
		Val Seen	**Val Unseen**	**Abs Gap** $\lvert\Delta\rvert$
Room-to-Room (Anderson et al., 2018b)	R2R (Anderson et al., 2018b)	38.6	21.8	16.8
	RPA (Wang et al., 2018b)	42.9	24.6	18.3
	S-Follower (Fried et al., 2018)	66.4	35.5	30.9
	RCM (Wang et al., 2019b)	66.7	42.8	23.9
	SMNA (Ma et al., 2019a)	67	45	22
	Regretful (Ma et al., 2019b)	69	50	19
	EnvDrop (Tan et al., 2019)	62.1	52.2	9.9
	ALTR (Huang et al., 2019)	55.8	46.1	9.7
	RN+Obj (Hu et al., 2019)	59.2	39.5	19.7
	Chasing Ghost (Anderson et al., 2019)	31	31	**0**
	Our baseline	56.1	47.5	8.6
	Our learned-semantic	53.1	**53.3**	**0.2**
Room-for-Room (Jain et al., 2019)	Speaker-Follower	51.9	23.8	28.1
	RCM	55.5	28.6	26.9
	Our baseline	54.6	30.7	23.9
	Our learned-semantic	36.2	**36.1**	**0.1**
CVDN (Thomason et al., 2019b)	NDH	5.92	2.10	3.82
	Our baseline	5.97	2.23	3.74
	Our learned-semantic	2.60	**2.43**	**0.17**
Touchdown (Chen et al., 2019c)	GA (original split)	7.9 (dev)	5.5 (test)	–
	RCONCAT (original split)	9.8 (dev)	10.7 (test)	–
	Our baseline (original split)	15.0 (dev)	14.2 (test)	–
	Our baseline (seen/unseen split)	17.5	5.3	12.2

formance on training environments and unseen testing environments: validation seen and validation unseen. The validation seen split takes the data from training environments while the validation unseen uses new environments besides the training environments.

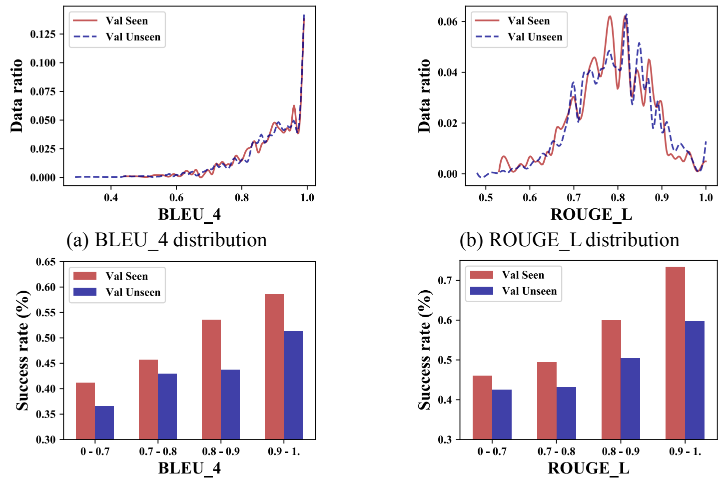
As shown in the first part of Table 1, we list most of the previous works on the Room-to-Room dataset (Anderson et al., 2018b) and report the *success rate* under greedy decoding (i.e., without beam-search) on validation seen and validation unseen splits. The large absolute gaps (from 30.9% to 9.7%) between the results from seen and unseen environments show that current neural agent models on R2R suffer from environment bias.[2] Besides Room-to-Room (R2R), we also analyze two newly-released indoor navigation datasets that were also collected from Matterport3D environments: Room-for-Room (R4R) (Jain et al., 2019) and Cooperative Vision-and-Dialog Navigation (CVDN) (Thomason et al., 2019b). As shown in the second and third part of Table. 1, models also suffer from the environment bias in these datasets with a significant performance drop from seen to unseen environments (i.e., 26.9% on R4R and 3.74 on CVDN). Lastly, we show the results (denoted as 'ours' in Table. 1) when the environment bias (reason analyzed in Sec. 5) is effectively eliminated by our learned semantic features (described in Sec. 6.3). As a result, the performance gaps are effectively decreased on all three datasets without changing the model and learning hyper-parameters, compared to our baselines (denoted as 'Our baseline') and previous works[3].

3.3 Environment Bias in Outdoor Navigation

The three indoor navigational datasets in previous sections are collected from the Matterport3D environments (Chang et al., 2017). In order to show that the environment bias also exists in other

[2]Our work's aim is to both close the seen-unseen gap while also achieving competitive unseen results. Note that Anderson et al. (2019) also achieve 0% gap but at the trade-off of low unseen results. There is also another recent work by Ke et al. (2019) but they do not report val-seen results for non-beam-search methods.

[3]As for another major evaluation metric on the R4R dataset, Coverage weighted by Length Score (CLS), we also observe a similar phenomenon in performance gap; and our methods can also eliminate this gap from 19.2 to 1.5 and achieve competitive state-of-the-art unseen CLS results (34.7).



(a) BLEU_4 distribution (b) ROUGE_L distribution

(c) Success rate w.r.t. BLEU_4 score (d) Success rate w.r.t. ROUGE_L score

Figure 2: The 'distance' (defined by language scores) distribution and its relationship to success rate on val seen and val unseen.

kinds of environment, we investigate an outdoor navigation dataset, Touchdown (Chen et al., 2019c), which takes New York City as its navigational environment. In the original data splits of Touchdown, the environment is not specifically divided into seen and unseen and only involved one city. Thus the trained agent is only tested on the training environments (similar to validation seen split). To reveal the environment bias in Touchdown dataset, we split the city environment according to latitude and create two sub-environments: 'training' and 'unseen'. The data are then re-split into training, val-seen, and val-unseen, accordingly. We adapt our baseline R2R agent model with additional convolutional layers to fit this new task. As shown in the last part of Table. 1, when experimenting on the original data split, our baseline model achieves state-of-the-art results on the original 'dev' set and 'test' set, proving the validity of our model in this dataset. However, the results on our re-split data (denoted as 'Our baseline (seen/unseen split)') still have a big drop from the 'training' to the 'unseen' sub-environment (from 17.5% to 5.3%), showing that environment bias is a broad issue.

4 WHERE: THE EFFECT OF DIFFERENT TASK COMPONENTS

In Sec. 3, we showed that current neural agent models are biased towards the training environments on multiple vision-and-language navigation (VLN) datasets. In this section, we want to locate which component of the VLN task is attributed to this environment bias. We use Room-to-Room (R2R) dataset (Anderson et al., 2018b) as a diagnosing dataset since it has been released for a while and hence well-explored by several previous works. We start by showing below that two possible candidates, the natural language instructions and the underlying navigational graph, do not directly contribute to the environment bias. We then analyze the effect of visual environments in detail.

4.1 THE EFFECT OF NATURAL-LANGUAGE NAVIGATIONAL INSTRUCTIONS

A common hypothesis is that the navigational instructions for unseen environments (e.g., val unseen) are much different from the training environments due to the different objects and layouts in new environments; and this lingual difference thus leads to the performance gap. In this section, we look into the instructions in val-seen (on the training environments) and val-unseen (on the unseen environments) split to verify this hypothesis. We analyze the success rate of the navigation instructions based on their relationships to training instructions. In order to quantitatively evaluate this relationship, we define the 'distances' from a validating instruction to all training instructions as the phrase-matching metric. Suppose x is a validating datum, $\mathbb{T}$ is the training set, and $\mathrm{inst}(x)$ is the instruction of the datum x, we use ROUGE-L (Lin, 2004) and corpus-level BLEU-4 Papineni

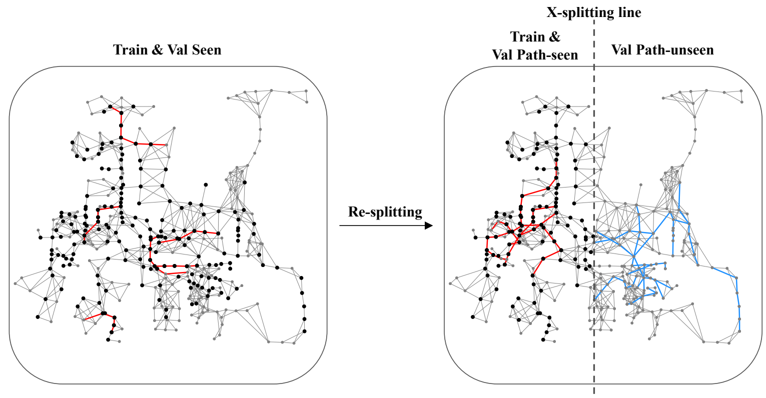


Figure 3: Graph split: black vertices are viewpoints visited during training; red paths are val seen / val path-seen; blue paths are val path-unseen.

et al. (2002) to calculate this 'distance':

$$\mathrm{dis}_{\text{ROUGE}}(x, \mathbb{T}) = \min_{t \in \mathbb{T}} \text{ROUGE-L}\left(\mathrm{inst}(x), \mathrm{inst}(t)\right) \quad (1)$$

$$\mathrm{dis}_{\text{BLEU}}(x, \mathbb{T}) = \text{BLEU-4}\left(\mathrm{inst}(x), \{\mathrm{inst}(t)\}_{t \in \mathbb{T}}\right) \quad (2)$$

where we consider all the training instructions as references in calculating the BLEU-4 score.

We show the statistics and results of instructions in two validation splits in Fig. 2. Different from the hypothesis, we did not observe significant difference between the distributions of 'distances' (as shown in Fig. 2) on seen validation and unseen validation. For the results, we find that the performance is better on instructions with smaller 'distances' (higher metric score w.r.t. the training instructions) for each validation split. However, with the same 'distance' to training instructions, success rate on seen validation still significantly outperforms the results on unseen validation, which implies the existence of other reasons rather than language attributed to this performance gap.

4.2 The Effect of Underlying Navigational Graph

As shown in Fig. 3, an environment could be considered as its underlying navigational graph with visual information (as in Fig. 1). In order to test whether the agent model could overfit to these navigational graphs (and thus be biased towards training environments), we follow the experiments in Hu et al. (2019) to train the agent without visual information. Specifically, we mask out the ResNet features with zero vectors thus the agent could only make decision based on the instructions and the navigational graph. With our baseline model, the success rate is 38.5% on validation seen and 41.0% on validation unseen, which is consistent with the finding in Hu et al. (2019). Besides showing the high results without visual contents (similar to Thomason et al. (2019a) and Hu et al. (2019)), we also want to emphasize the low performance gap between seen and unseen environments (2.5% compared to the $> 10\%$ gap in usual). Hence, we claim that the underlying graph is not a necessary reason to the environment bias.

4.3 The Effect of Visual Environments

To show how the visual environments affect the agent's performance, we analyze the agent's results on unseen environments and in different regions of the training environments. In order to give a detailed characterization of the effect of environments, we are going to reveal spatial localities of the agent's performance in environments at three different levels:

- **Path-level Locality:** Agents are better at paths which intersect with the training paths.
- **Region-level Locality:** Agents are better in regions which are closer to the training data.
- **Environment-level Locality:** Agents perform better on training environments than on unseen environments.

Table 2: Results on our re-splitting data showing the path-level and environment-level localities.

	Splitting Method	Train	Validation		
			Path-seen	Path-unseen	Env-unseen
Environments	R2R	61	56	0	11
	X-split	61	57	16	11
	Z-split	61	56	29	11
Number of Data	R2R	14,025	1,020	0	2,349
	X-split	11,631	1,230	1,098	2,349
	Z-split	10,894	867	2,324	2,349
Success Rate	R2R	–	56.1	–	47.5
	X-split	–	58.9	52.6	46.7
	Z-split	–	62.5	47.8	42.4

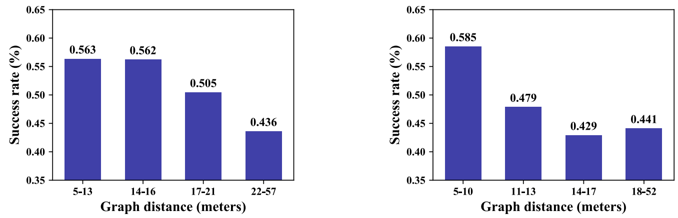
And the existence of these spatial locality inspires us to find the direct cause in Sec. 4.1. However, the original split of data is not fine-grained enough to reveal these spatial localities. To better illustrate this, we visualize the data from one environment of the Room-to-Room dataset in Fig. 3, where the vertices are viewpoints with visual information and edges are valid connections between viewpoints. The vertices highlighted with dark-black indicate the viewpoints which are used in training paths. And the red edges are the connections covered by original val-seen paths. As shown in Fig. 3, nearly all viewpoints in val-seen paths (vertices connected to red lines) are used as viewpoints in training data (vertices marked by dark-black). We thus cannot categorize the path-level and region-level localities. To bypass this, we propose a novel re-splitting method to create our diagnosis data splits.

Structural Data Re-splitting We employ two kinds of structural data splitting methods based on the horizontal or vertical coordinates, denoted as 'X-split' and 'Z-split', respectively. The 'Z-split' intuitively separates different floors in the houses and 'X-split' creates separate areas. When applying to the training environments in R2R dataset, we use one side of the splitting line (see the 'X-splitting line' Fig. 3) as the new training 'environment', and the other side as the path-unseen 'environment'. In addition to this split of environments, we also re-split the original training data and val-seen data while keeping the val-unseen data the same. The data paths across the splitting line are dropped. As shown in the right part of Fig. 3, we create three new data splits: training split, val-path-seen split, and val-path-unseen split. The edges covered by the new val-path-unseen split are highlighted in red, while the color style of training split and val-path-seen split ('Black' for viewpoints in training and 'Red' for edges in val-path-seen) are the same. Since the number of original val-seen data are not enough to fill two new validation sets (val-path-seen and val-path-unseen), we bring some (original) training data into our new validation splits. The overall statistics of original splits and our new splits are shown in Table 2.[4]

Existence of Path-level and Environment-level Localities For both splitting methods, we train our baseline model on the newly-split training set and evaluate on our three validation sets (denoted as 'X-split' or 'Z-split' rows in Table 2). We also show the results of our baseline model on the original R2R (denoted as 'R2R' rows) splits for a comparison. As shown in Table. 2, the results in val path-seen is higher than val path-unseen. It suggest that a **path-level locality** exits in current VLN agent models . Meanwhile, the results on val path-unseen are further higher than val env-unseen and it shows the **environment-level locality** which is independent of the path-level locality.

Existence of the Region-level Locality To further demonstrate region-level locality, we study how the success rate changes in different regions of the environment with respect to their distances to the training data, which is similar to the analysis of language 'distance' in Sec. 4.1. We first calculate the point-by-point shortest paths using the Dijkstra's algorithm (Dijkstra, 1959), where the shortest distances between viewpoints v and v' are denoted as the graph distance $\mathrm{dis}_{\mathrm{GRAPH}}(v, v')$. Based on this graph distance, we define the viewpoint-distance $\mathrm{dis}_{\mathrm{VIEWPOINT}}$ from a viewpoint v to the training data $\mathbb{T}$ as v's minimal graph distance to a viewpoint v' in training data. We then define

[4]We only split the environments whose data contains substantial amount, thus make sure that the remaining training data is still adequate for training strong models.



(a) X-split (b) Z-split

Figure 4: The success rate declines as the path moves further from training regions.

the path-distance $\mathrm{dis}_{\mathrm{PATH}}$ from a validating data x and the whole training data $\mathbb{T}$ as the maximal viewpoint-distance in the path of x:

$$\mathrm{dis}_{\mathrm{PATH}}(x, \mathbb{T}) = \max_{v \in \mathrm{path}(x)} \mathrm{dis}_{\mathrm{VIEWPOINT}}(v, \mathbb{T}) \tag{3}$$

$$= \max_{v \in \mathrm{path}(x)} \left\{ \min_{\substack{v' \in \mathrm{path}(t) \\ \forall t \in \mathbb{T}}} \mathrm{dis}_{\mathrm{GRAPH}}\left(v, v'\right) \right\} \tag{4}$$

We compute this path-distance between paths in the env-seen validation set and training environments in our re-split data. As shown in Fig. 4, the success rate declines as the path moves further from training environment on both re-splitting methods (i.e., ‘X-split’ and ‘Z-split’). As a conclusion, the closer the path to the training data, the higher the agent performance is.

5 WHY: WHAT INSIDE THE ENVIRONMENTS IS ATTRIBUTED TO THE BIAS?

In Sec. 4, we locate that the performance gap is due to the visual environments by excluding other reasons and categorizing the spatial localities. However, it still remains multiple possible aspects inside the environment which could lead to these spatial localities, e.g., the object layout convention and the room connections. The agent model could be biased towards the training environments by over-fitting or memorizing these environment-specific characteristics. In this section, we want to identify which aspect directly contributes to the bias and suggest a surprising conclusion: the environment bias is attributed to low-level visual information carried by the ResNet features.

We first show an experiment which effectively decreases the gap between seen and unseen environments with minimal model modifications. We then clarify our conclusions based on the findings.

5.1 AN INVESTIGATION EXPERIMENT: IMAGENET LABELS AS VISUAL FEATURES

Suspecting that the over-fitting happens when the agent over-learns low-level features, we hope to find the replacement of ResNet 2048-features that contain minimal low-level information while preserving distinguishable visual contents. The most straightforward replacement is that instead of using mean-pooled features, we inject the frozen 1000-way classifying layer in ResNet pre-training, and use the probabilities of ImageNet labels as visual features. Shown as ‘ImageNet’ in Table. 3, the probability distribution almost closes the gap between seen and unseen. These results further constrain the reason of environment bias to the low-level ResNet features of image views. Combining with the findings of spatial localities, we suggest that environments (i.e., houses) and regions (i.e., rooms) usually have their own ‘style’. Thus the same semantic label (captured by ImageNet-1000 features) has different visual appearances (captured by ResNet features) in different environments or regions. As a result, ImageNet-1000 features, in spite of being noisy, are not distracted by low-level visual appearance and could generalize to unseen environments, while ResNet features could not.

Although these ImageNet-1000 features decrease the performance gap, it has a disagreement to the VLN domain so that the validation unseen results of R4R and CVDN are slightly worse than baseline (and not much better for R2R). Hence it motivates us to find the better semantic representations of environmental features that can both close the seen-unseen gap while also achieving state-of-the-art on unseen results (which we discuss next).

Table 3: Results showing that our semantic feature representations eliminate the performance gap, which further proves the over-fitting located in visual environments.

Task	Feature			Result		
	Type	**Name**	**Dim**	**Val Seen**	**Val Unseen**	**Abs Gap $\lvert\Delta\rvert$**
Room-to-Room	Baseline	ResNet NoDrop	$2,048$	54.5	38.2	16.3
	Baseline	ResNet	$2,048$	56.1	47.5	8.6
	Invesgation	ImageNet	$1,000$	47.1	48.2	1.1
	Semantic	Detection	152	55.9	50.0	5.9
	Semantic	Ground Truth	42	55.6	56.2	0.6
	Semantic	Learned	42	53.1	53.3	0.2
R4R	Baseline	ResNet NoDrop	$2,048$	52.5	25.8	26.7
	Baseline	ResNet	$2,048$	54.6	30.7	23.9
	Investigation	ImageNet	$1,000$	28.7	28.9	0.2
	Semantic	Detection	152	48.8	32.0	16.8
	Semantic	Ground Truth	42	47.6	35.9	11.7
	Semantic	Learned	42	36.2	36.1	0.1
CVDN	Baseline	ResNet NoDrop	$2,048$	5.88	2.14	3.74
	Baseline	ResNet	$2,048$	5.97	2.23	3.74
	Invesgation	ImageNet	$1,000$	3.22	2.08	1.14
	Semantic	Detection	152	3.34	2.08	1.26
	Semantic	Ground Truth	42	3.75	2.69	1.06
	Semantic	Learned	42	2.60	2.43	0.17

6 How: Methodology to Fix the Environment Bias

In the previous section (Sec. 5), we found that the reason for the environmental bias is related to the low-level visual features (i.e., 2048-dim ResNet features). Following the findings we observe in Sec. 5.1, we build our agent on the features which are more correlated to the VLN environmental semantics than the ImageNet label features in Sec. 5.1. We first demonstrate our baseline results on three VLN datasets and then explore the advanced semantic feature replacements. As shown in Table 3, these advanced semantic features could effectively reduce the performance gap between seen and unseen environments and improve the unseen results compared to our strong baselines. The effectiveness of these semantic features supports our explanation of the environment bias in Sec. 5 and also suggests that future work in VLN tasks should think about such generalization issues.

6.1 Baseline

In our baseline model, following the previous works we use the standard ResNet features as the representation of environments (Anderson et al., 2018b; Jain et al., 2019; Thomason et al., 2019b). These features come from the mean-pooled layer after the final convolutional layer of ResNet-152 (He et al., 2016) pre-trained on ImageNet (Russakovsky et al., 2015). As the 'Baseline'[5] rows in Table. 3, the phenomenon of environment bias is shown in all three datasets, where val-seen results are significantly higher than val-unseen results. Note that our baseline method take the 'feature dropout' technique demonstrated in Tan et al. (2019) (without back translation): the ResNet features are randomly masked by zero before using as inputs of the agent. Without this 'feature dropout' (denoted as 'ResNet NoDrop' in Table. 3), the gaps will increase in R2R and R4R, which suggests that this 'feature dropout' technique also helps to eliminate over-fitting the low-level visual information conveyed by ResNet features as we discussed in Sec. 5. However, the performance gap is still quite large, which leads us to the following discussions of semantic features.

6.2 Detected Objects Areas

During the navigation, the objects in the environments are crucial since their matchings with the instruction often indicate the locations that can guide the agent. Thus the object detection results

[5]We use our baseline agent model (in Sec. 3.1) for R2R and R4R. For CVDN, we take the official baseline code in https://github.com/mmurray/cvdn.

from the environments can provide relevant semantic information. In our work, we utilize the detection information generated by Faster R-CNN (Ren et al., 2015) to create the feature representations. Comparing to the ImageNet-1000 features (Sec. 5.1), this detection feature needs to include more environmental information since the viewing images in VLN usually contains multiple objects. Instead of using the classification probability of the labels from ResNet, we design our detection features $\mathrm{f}_{\mathrm{DETECT}}$ of every image view as the sum of the areas of detected objects weighted by the detection confidence, which is different from the approach in Hu et al. (2019) who utilized the detection results by using the embeddings of detected labels:

$$\mathrm{f}_{\mathrm{DETECT}}=[a_{c_1}, a_{c_2}, \ldots, a_{c_n}]; \qquad a_{c_i}=\sum_{\text{obj is } c_i} \mathrm{Area(obj)} \cdot \mathrm{Conf(obj)} \tag{5}$$

where the c_i and a_{c_i} are the label and feature of each detected object, $\mathrm{Area}(*)$ and $\mathrm{Conf}(*)$ are the area and confidence of each object. For implementation details, we use the Faster R-CNN (Ren et al., 2015) trained on Visual Genome (Krishna et al., 2017) provided in Bottom-Up Attention (Anderson et al., 2018a). To eliminate the the labels irrelevant to VLN task, we calculate the total areas of each detection objects among all environments and pick the labels that take up a relatively large proportion of the environments, creating features of dimension 152.[6] Denoted as 'Detection' in Table 3, the performance gap is diminished with this detection features compared to baselines in all three datasets, indicating that changing the features to a higher semantic level has positive effect on alleviating the environment bias. Meanwhile, the improvement of unseen validation results on R2R an R4R datasets suggests the better efficiency in the VLN task than the ImageNet labels.

6.3 Semantic Segmentation

Although the detection features can provide enough semantic information for the agent to achieve comparable results as baseline model, they does not fully utilize the full environmental information, ignoring the visual content left over from detection that may contain useful knowledge during navigation. A better semantic representation is the semantic segmentation, which segments each view image on pixel-wise and gives the label to each segment region, allowing us to utilize the semantics from the entire environment. Matterport3D (Chang et al., 2017) dataset provides the labeled semantic segmentation information of every scene and we take the rendered images from Tan et al. (2019)[7]. A comparison example of RGB images and semantic views are available in Appendix. Since the semantic segmentation images are fine-grained and blurry in boundaries, we follow the design of detection features, using the areas of semantic classes in each image view as the semantic features (confidence is excluded since semantic segmentation does not provide this value). We first assume that the semantic information is provided as additional environmental information and the results of the model using the ground truth semantic areas are shown in the 'ground truth' rows in Table. 3. We next train a separate multi-layer perceptron to predict the areas of these semantic labels, and the results of the model with these predicted semantics as features are shown in 'learned'. As shown in Table. 3, both 'ground truth' and 'learned' semantic feature representations bring the performance of seen and unseen closer comparing to baseline model, and the smallest performance gaps come from learned semantic segmentation features in all three datasets. The highest validation unseen success rates among all the proposed feature representations are also produced by semantic segmentation features, 'learned' semantic for R4R and 'ground truth' semantic for R2R and CVDN. Overall, among all the semantic representations we have explored, the semantic segmentation features are most effective in eliminating the environment bias.

7 Conclusion

In this paper, we focus on studying the performance gap between seen and unseen environments widely observed in vision-and-language navigation (VLN) tasks, trying to find where and why this environment bias exists and provide possible initial solutions. By designing the diagnosis experiments of environment re-splitting and feature replacement, we locate the environment bias to be in the low-level visual appearance; and we discuss semantic features that decrease the performance gap in three VLN datasets and achieve state-of-the-art results.

[6]Note that this detection feature dimension 152 coincidentally is the same as the number of layers in ResNet, but there is no correlation.

[7]The rendered semantic views are downloaded from https://github.com/airsplay/R2R-EnvDrop.

REFERENCES

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6077–6086, 2018a.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3674–3683, 2018b.

Peter Anderson, Ayush Shrivastava, Devi Parikh, Dhruv Batra, and Stefan Lee. Chasing ghosts: Instruction following as bayesian state tracking. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *The International Conference on Learning Representations (ICLR)*, 2015.

Gilles Blanchard, Gyemin Lee, and Clayton Scott. Generalizing from several related classification tasks to a new unlabeled sample. In *Advances in neural information processing systems*, pp. 2178–2186, 2011.

Gilles Blanchard, Aniket Anand Deshmukh, Urun Dogan, Gyemin Lee, and Clayton Scott. Domain generalization by marginal transfer learning. *arXiv preprint arXiv:1711.07910*, 2017.

Valts Blukis, Dipendra Misra, Ross A Knepper, and Yoav Artzi. Mapping navigation instructions to continuous control actions with position-visitation prediction. In *Conference on Robot Learning*, pp. 505–518, 2018.

Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2229–2238, 2019.

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

Chao Chen, Zhihong Chen, Boyuan Jiang, and Xinyu Jin. Joint domain alignment and discriminative feature learning for unsupervised deep domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3296–3303, 2019a.

Chaoqi Chen, Weiping Xie, Wenbing Huang, Yu Rong, Xinghao Ding, Yue Huang, Tingyang Xu, and Junzhou Huang. Progressive feature alignment for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 627–636, 2019b.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 12538–12547, 2019c.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 2054–2063, 2018.

Aniket Anand Deshmukh, Yunwen Lei, Srinagesh Sharma, Urun Dogan, James W Cutler, and Clayton Scott. A generalization error bound for multi-class domain generalization. *arXiv preprint arXiv:1905.10392*, 2019.

Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1 (1):269–271, 1959.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *Advances in Neural Information Processing Systems*, pp. 3314–3325, 2018.

Rui Gong, Wen Li, Yuhua Chen, and Luc Van Gool. Dlow: Domain flow for adaptation and generalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2477–2486, 2019.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. Iqa: Visual question answering in interactive environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4089–4098, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Ehsan Hosseini-Asl, Yingbo Zhou, Caiming Xiong, and Richard Socher. Augmented cyclic adversarial learning for low resource domain adaptation. In *The International Conference on Learning Representations (ICLR)*, 2019.

Ronghang Hu, Daniel Fried, Anna Rohrbach, Dan Klein, Kate Saenko, et al. Are you looking? grounding to multiple modalities in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Haoshuo Huang, Vihan Jain, Harsh Mehta, Alexander Ku, Gabriel Magalhaes, Jason Baldridge, and Eugene Ie. Transferable representation learning in vision-and-language navigation. In *IEEE International Conference on Computer Vision*, 2019.

Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2019.

Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6741–6749, 2019.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5542–5550, 2017.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.

Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pp. 1640–1650, 2018.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *The International Conference on Learning Representations (ICLR)*, 2019a.

Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided navigation through progress estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6732–6740, 2019b.

Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. Walk the talk: connecting language, knowledge, and action in route instructions. In *proceedings of the 21st national conference on Artificial intelligence-Volume 2*, pp. 1475–1482. AAAI Press, 2006.

Piotr Mirowski, Matt Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Andrew Zisserman, Raia Hadsell, et al. Learning to navigate in cities without a map. In *Advances in Neural Information Processing Systems*, pp. 2419–2430, 2018.

Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. Mapping instructions to actions in 3d environments with visual goal prediction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2667–2678, 2018.

Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International Conference on Machine Learning*, pp. 10–18, 2013.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318. Association for Computational Linguistics, 2002.

Yuankai Qi, Qi Wu, Peter Anderson, Marco Liu, Chunhua Shen, and Anton van den Hengel. Rerere: Remote embodied referring expressions in real indoor environments. *arXiv preprint arXiv:1904.10151*, 2019.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91–99, 2015.

Artem Rozantsev, Mathieu Salzmann, and Pascal Fua. Residual parameter transfer for deep domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4339–4348, 2018.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 2610–2621, 2019.

Jesse Thomason, Daniel Gordan, and Yonatan Bisk. Shifting the baseline: Single modality performance on visual navigation & qa. In *NACCL*, 2019a.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *CoRL*, 2019b.

Daixin Wang, Peng Cui, and Wenwu Zhu. Deep asymmetric transfer network for unbalanced domain adaptation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018a.

Ximei Wang, Liang Li, Weirui Ye, Mingsheng Long, and Jianmin Wang. Transferable attention for domain adaptation. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2019a.

Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 37–53, 2018b.

Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6629–6638, 2019b.

Licheng Yu, Xinlei Chen, Georgia Gkioxari, Mohit Bansal, Tamara L Berg, and Dhruv Batra. Multi-target embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6309–6318, 2019.

Yabin Zhang, Hui Tang, Kui Jia, and Mingkui Tan. Domain-symmetric networks for adversarial domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5031–5040, 2019.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232, 2017.

A Appendix

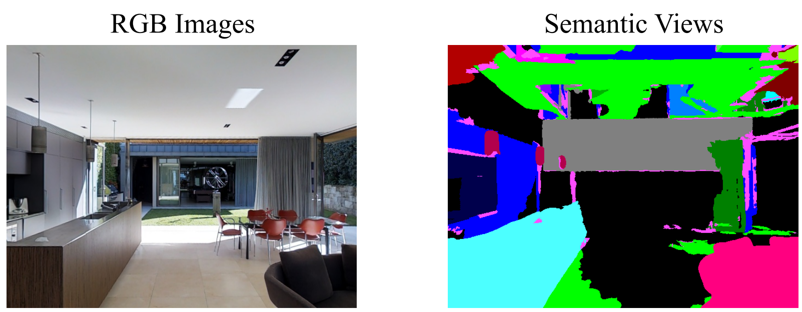


Figure 5: Comparisons between RGB images and their semantic views.

A.1 Examples of RGB Images and Semantic Views

In Fig. 5, we show a rendered semantic view from Tan et al. (2019) and its original RGB image. Different colors indicate different semantic segmentation areas and 40 semantic labels are considered in the Matterport3D dataset Chang et al. (2017).