Airbert: In-domain Pretraining for Vision-and-Language Navigation

Pierre-Louis Guhur, Makarand Tapaswi, Shizhe Chen, Ivan Laptev, Cordelia Schmid

Airbert: In-domain Pretraining for Vision-and-Language Navigation

Pierre-Louis Guhur[1], Makarand Tapaswi[2], Shizhe Chen[1], Ivan Laptev[1], Cordelia Schmid[1]
[1]Inria, École normale supérieure, CNRS, PSL Research University, Paris, France
[2]IIIT Hyderabad, India
pierre-louis.guhur@inria.fr https://airbert-vln.github.io

Abstract

Vision-and-language navigation (VLN) aims to enable embodied agents to navigate in realistic environments using natural language instructions. Given the scarcity of domain-specific training data and the high diversity of image and language inputs, the generalization of VLN agents to unseen environments remains challenging. Recent methods explore pretraining to improve generalization, however, the use of generic image-caption datasets or existing small-scale VLN environments is suboptimal and results in limited improvements. In this work, we introduce BnB[1], a large-scale and diverse in-domain VLN dataset. We first collect image-caption (IC) pairs from hundreds of thousands of listings from online rental marketplaces. Using IC pairs we next propose automatic strategies to generate millions of VLN path-instruction (PI) pairs. We further propose a shuffling loss that improves the learning of temporal order inside PI pairs. We use BnB to pretrain our Airbert[2] model that can be adapted to discriminative and generative settings and show that it outperforms state of the art for Room-to-Room (R2R) navigation and Remote Referring Expression (REVERIE) benchmarks. Moreover, our in-domain pretraining significantly increases performance on a challenging few-shot VLN evaluation, where we train the model only on VLN instructions from a few houses.

1. Introduction

In vision-and-language navigation (VLN), an agent is asked to navigate in home environments following natural language instructions [3, 5]. This task is attractive to many real-world applications such as domestic robotics and personal assistants. However, given the high diversity of VLN data across environments and the difficulty of the manual collection and annotation of VLN training data at scale, the performance of current methods remains limited, especially for previously unseen environments [49].

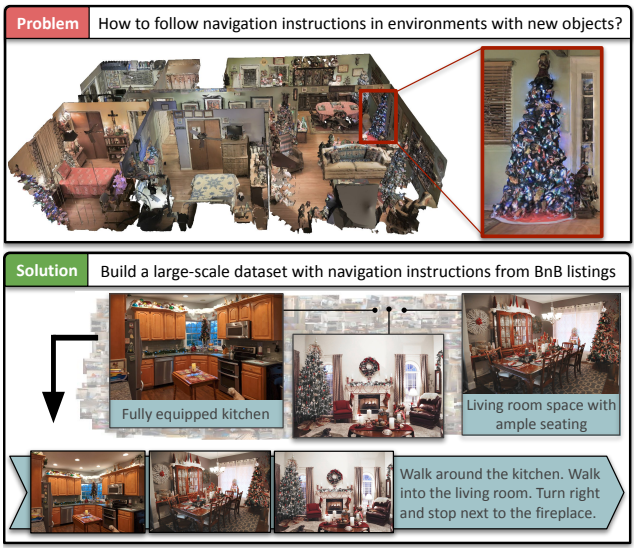


Figure 1: VLN tasks are evaluated on unseen environments at test time. *Top*: None of the training houses contain a Christmas theme making this test environment particularly challenging. *Bottom*: We build a large-scale, visually diverse, and in-domain dataset by creating path-instruction pairs close to a VLN-like setup and show the benefits of self-supervised pretraining.

Our work is motivated by significant improvements in vision and language pretraining [2, 9, 23, 24, 25, 38], where deep transformer models [42] are trained via self-supervised proxy tasks [10] using large-scale, automatically harvested image-text datasets [32, 36]. Such pretraining enables learning transferable multi-modal representations achieving state-of-the-art performance in various vision and language tasks. Similarly, with the goal of learning an embodied agent that generalizes, recent works [13, 16, 22, 28] have explored different pretraining approaches for VLN tasks.

In [13, 16], annotated path-instruction pairs are augmented with a *speaker* model that generates instructions for random unseen paths. However, as these paths originate from a small set of 61 houses used during training, they are limited in visual diversity. The limited pretraining

[1]Bed and Breakfast
[2]*Airbert* is an Old Irish word meaning *practice*, here referring to model pretraining on pretext tasks similar to VLN.

environments do not equip agents with visual understanding abilities that enable generalization to unseen houses, see Fig. 1. To address this problem, VLN-BERT [28] proposes to pretrain the agent on generic image-caption datasets that are abundant and cover diverse visio-linguistic knowledge. However, these image-caption pairs are quite different from the dynamic visual stream (path) and navigable instructions observed by a VLN agent. Such out-of-domain pretraining, although promising, only brings limited gains to the navigation performance. Besides the above limitations, existing pretraining methods do not place much emphasis on temporal reasoning abilities in their proxy tasks such as one-step action prediction [13] and path-instruction pairing [28], while such reasoning is important to a sequential decision making task like VLN. As a result, even if performance in downstream tasks is improved, the pretrained models may still be brittle. For example, a simple corruption of instructions by swapping noun phrases within the instruction, or replacing them with other nouns, leads to significant confusion as models are unable to pick the correct original pair.

In this paper, we explore a different data source and proxy tasks to address the above limitations in pretraining a generic VLN agent. Though navigation instructions are rarely found on the Internet, image-caption pairs from home environments are abundant in online marketplaces (*e.g. Airbnb*), which include images and descriptions of rental listings. We collect BnB, a new large-scale dataset with 1.4M indoor images and 0.7M captions. First, we show that in-domain image-caption pairs bring additional benefits for downstream VLN tasks when applied with generic web data [28]. In order to further reduce the domain gap between the BnB pretraining and the VLN task, we present an approach to transform static image-caption pairs into visual paths and navigation-like instructions (Fig. 1 bottom), leading to large additional performance gains. We also propose a shuffling loss that improves the model's temporal reasoning abilities by learning a temporal alignment between a path and the corresponding instruction.

Our pretrained model, Airbert, is a generic transformer backbone that can be readily integrated in both discriminative VLN tasks such as path-instruction compatibility prediction [28] and generative VLN tasks [15] in R2R navigation [5] and REVERIE remote referring expression [34]. We achieve state-of-the-art performance on these VLN tasks with our pretrained model. Beyond the standard evaluation, our in-domain pretraining opens an exciting new direction of *one/few-shot VLN* where the agent is trained on examples only from one/few environment(s) and expected to generalize to other unseen environments.

In summary, the contributions of this work are threefold. (1) We collect a new large-scale in-domain dataset, BnB, to promote pretraining for vision-and-language navigation tasks. (2) We curate the dataset in different ways to reduce the distribution shift between pretraining and VLN and also propose the shuffling loss to improve temporal reasoning abilities. (3) Our pretrained Airbert can be plugged into generative or discriminative architectures and achieves state-of-the-art performance on R2R and REVERIE datasets. Moreover, our model generalizes well under a challenging one/few-shot VLN evaluation, truly highlighting the capabilities of our learning paradigm.

2. Related work

Vision-and-language navigation. VLN [5] has received significant attention with a large number of followup tasks introduced in recent years [3, 8, 19, 21, 30, 31, 34, 37, 41]. Early days of VLN saw the use of sequence-to-sequence LSTMs to predict low-level actions [5] or high-level directions in a panoramic action space [11]. Different attention mechanisms [26, 33] are proposed to improve cross-modal alignment. Various reinforcement learning based training algorithms [40, 45, 47, 48] and searching algorithms in inference [11, 26, 27] have also been explored to improve the VLN performance.

To improve an agent's generalization to unseen environments, data augmentation is performed by using a *speaker* model [11] that generates instructions for random paths in seen environments, and environment dropout [40] is used to mimic new environments. While pretraining LSTMs for transferable representations is adopted by [16], recently, there has been a shift towards transformer models [13] to learn generic multimodal representations. This is further extended to a recurrent model that significantly improves sequential action prediction [15]. However, the limited environments in pretraining [13, 16] constrain the generalization ability to unseen scenarios. Most related to this work, VLN-BERT [28] transfers knowledge from abundant, but out-of-domain image-text data to improve path-instruction matching. In this work, we not only create a large-scale, *in-domain* BnB dataset, but also propose effective pretraining strategies to mitigate the domain-shift between webly crawled image-text pairs and VLN data.

Large-scale visio-linguistic pretraining. Thanks to large-scale image-caption pairs automatically collected from the web [29, 32, 35, 36], visio-linguistic pretraining (VLP) has made great breakthroughs in recent years. Several VLP models [9, 23, 24, 39] have been proposed based on the transformer architecture [42]. These models are often pretrained with self-supervised objectives akin to those in BERT [10]: masked language modeling, masked region modeling and vision-text pairing. Fine-tuning them on downstream datasets achieves state-of-the-art performance on various VL tasks [6, 18, 46, 44]. While such pretraining focuses on learning correlations between vision and text, it is not designed for sequential decision making as required

in embodied VLN. The goal of this work is not to improve VLP architectures but to present in-domain training strategies that lead to performance improvements for VLN tasks.

3. BnB Dataset

Hosts that rent places on online marketplaces often upload attractive and unique photos along with descriptions. One such marketplace, *Airbnb*, has 5.6M listings from over 100K cities all around the world [1]. We propose to use this abundant and curated data for large-scale in-domain VLN pretraining. In this section, we first describe how we collect image-caption pairs from *Airbnb*. Then, we propose methods to transform images and captions into VLN-like path-instruction pairs to reduce the domain gap between webly crawled image-caption pairs and VLN tasks (see Fig. 2).

3.1. Collecting BnB Image-Caption Pairs

Collection process. We restrict our dataset to listings from the US (about 10% of *Airbnb*) to ensure high quality English captions and visual similarity with Matterport environments [7]. The data collection proceeds as follows: (1) obtain a list of locations from Wikipedia; (2) find listings in these locations by querying the *Airbnb* search engine; (3) download listings and their metadata; (4) remove *outdoors* images[3] as classified by a ResNet model pretrained on Places365 [50]; and (5) remove invalid image captions such as emails, URLs and duplicates.

Statistics. We downloaded almost 150k listings and their metadata (1/4 of the listings in the US) in step 3, leading to over 3M images and 1M captions. After data cleaning with steps 4 and 5, we obtain 713K image-caption pairs and 676K images without captions. Table 1 compares our BnB dataset to other datasets used in previous works for VLN (pre-)training. It is larger than R2R [5], REVERIE [34] and includes a large diversity of rooms and objects, which is not the case for Conceptual Captions [36]. We posit that such in-domain data is crucial to deal with the data scarcity challenge in VLN environments as illustrated in Fig. 1. We use 95% of our BnB dataset for training and the remaining 5% for validation.

Apart from images and captions, our collected listings contain structured data including a list of amenities, a general description, reviews, location, and rental price, which may offer additional applications in the future. More details about the dataset and examples are presented in the Appendix A.

3.2. Creating BnB Path-Instruction Pairs

BnB image-caption (IC) pairs are complementary to Conceptual Captions (ConCaps) as they capture diverse VLN environments. However, they still have large differences from path-instruction (PI) pairs in VLN tasks. For example, during navigation, an agent observes a sequence of panoramic views rather than a single image, and the instruction may contain multiple sentences. To mitigate this domain gap, we propose strategies to automatically craft path-instruction pairs starting from BnB-IC pairs.

Dataset	Source	#Envs	#Imgs	#Texts
R2R [5]	Matterport	90	10.8K	21.7K
REVERIE [34]	Matterport	86	10.6K	10.6K
Speaker [40]	Matterport	60	7.8K	0.2M
ConCaps [36]	Web images	-	3.3M	3.3M
BnB (ours)	Airbnb	140K	1.4M	0.7M

Table 1: Comparing BnB to other existing VLN datasets. The #images from Matterport environments [7] refers to the #panoramas. The speaker model [40] generates instructions for randomly selected trajectories, but is limited to panoramas from 60 training environments. Note that the data from Conceptual Captions (ConCaps) may feature some houses, but it is not the main category.

3.2.1 Concatenating Images and Texts in a BnB Listing

Images in a BnB listing usually depict different locations in a house, mimicking the sequential visual observations an agent makes while navigating in the house. To create a VLN-like path-instruction pair, we randomly select and concatenate K[4] image-caption pairs from the listing. In between each caption, we randomly add a word from "*and*", "*then*", "." or nothing to make the concatenated instruction more fluent and diverse.

3.2.2 Augmenting *Paths* with Visual Contexts

In the above concatenated path, each location only contains one BnB image, and perhaps with a limited view angle as hosts may focus on objects or amenities they wish to highlight. Therefore, it lacks the panoramic visual context at each location that the agent receives in real navigation paths. Moreover, each location in the concatenated instruction is described by a unique sentence, while adjacent locations are often expressed together in one sentence in VLN instructions [14]. To address the above issues with concatenation, we propose two approaches to compose paths that have more visual context and can also leverage the abundant images without captions (denoted as *captionless images*).

1. Image merging extends the panoramic context of a location by grouping images from similar room categories (see Fig. 2). For example, if the image depicts a kitchen sink, it is natural to expect images of other objects such as forks and knives nearby. Specifically, we first cluster images of similar categories (*e.g. kitchen*) using room labels predicted by a

[3] While outdoor images may contain interesting features (*e.g.* a patio), we observe that removing them increases performance.

[4] typically 4 - 7 to match the number of steps in the R2R dataset

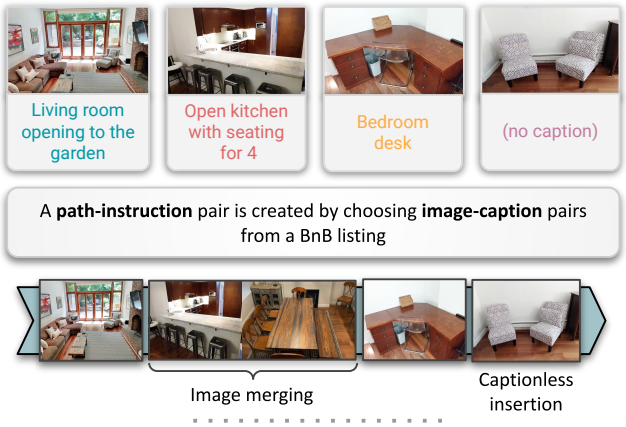


Figure 2: We explore several strategies to automatically create navigation-like instructions from image-caption pairs.

pretrained Places365 model [50]. Then, we extract multiple regions from this *merged* set of images, and use them as an approximation to the panoramic visual representation.

2. Captionless image insertion. Table 1 shows that half of the BnB images are captionless. Using them allows to increase the size of the dataset. When creating a path-instruction pair from the concatenation approach, a captionless image is inserted as if its caption was an empty string. The BnB PI pairs hence better approximate the distribution of the R2R path-instructions: (1) some images in the path are not described and (2) instructions have similar number of noun phrases.

3.2.3 Crafting *Instructions* with Fluent Transitions

The concatenated captions mainly describe rooms or objects at different locations, but do not contain any of the actionable verbs as in navigation instructions, *e.g.* "*turn left at the door*" or "*walk straight down the corridor*". We suggest two strategies to create fake instructions that have fluent transitions between sentences.

1. Instruction rephrasing. We use a fill-in-the-blanks approach to replace noun-phrases in human annotated navigation instructions [5] by those in BnB captions (see Fig. 2). Concretely, we create more than 10K instruction templates containing 2-7 blanks, and fill the blanks with noun-phrases extracted from BnB captions. The noun-phrases matched to object categories from the Visual Genome [20] dataset are preferred during selection. This allows us to create VLN-like instructions with actionable verbs interspersed with room and object references for visual cues that are part of the BnB path (see Fig. 2).

2. Instruction generation is a video captioning like model that takes in a sequence of images and generates an instruction corresponding to an agent's path through an environment. To train this model, we adopt ViLBERT and train it to generate captions for single BnB image-caption pairs. Further, this model is fine-tuned on trajectories of the R2R dataset to generate instructions. Finally, we use this model to generate BnB PI pairs by producing an instruction for a concatenated image sequence from BnB (the path).

4. Airbert: A Pretrained VLN Model

In this section, we present Airbert, our multi-modal transformer pretrained on the BnB dataset with masking and shuffling losses. We first introduce the architecture of Airbert, and then describe datasets and pretext tasks in pretraining. Finally, we show how Airbert can be adapted to downstream VLN tasks.

4.1. ViLBERT-like Architecture

ViLBERT [24] is a multi-modal transformer extended from BERT [10] to learn joint visio-linguistic representations from image-caption pairs, as illustrated in Fig. 3.

Given an image-caption pair (V, C), the model encodes the image as region features $[v_1, \ldots, v_\mathcal{V}]$ via a pretrained Faster R-CNN [4], and embeds the text as a series of tokens: $[\texttt{[CLS]}, w_1, \ldots, w_T, \texttt{[SEP]}]$, where `[CLS]` and `[SEP]` are special tokens added to the text. ViLBERT contains two separate transformers that encode V and C and it learns cross-modal interactions via co-attention [24].

We follow a similar strategy to encode path-instruction pairs (created in Sec. 3.2) that contain multiple images and captions $\{(V_k, C_k)\}_{k=1}^K$. Here, each V_k is represented as visual regions v_i^k and C_k as word tokens w_t^k. Respectively, the visual and text inputs to Airbert are:

$$X_V = [\texttt{[IMG]}, v_1^1, \ldots, v_{\mathcal{V}_1}^1, \ldots, \texttt{[IMG]}, v_1^K, \ldots, v_{\mathcal{V}_K}^K], \quad (1)$$

$$X_C = [\texttt{[CLS]}, w_1^1, \ldots, w_{T_1}^1, \ldots, w_1^K, \ldots, w_{T_K}^K, \texttt{[SEP]}], \quad (2)$$

where the `[IMG]` token is used to separate image region features taken at different locations.

Note that while our approach is not limited to a ViLBERT-like architecture, we choose ViLBERT for a fair comparison with previous work [28].

4.2. Datasets and Pretext Tasks for Pretraining

We use Conceptual Captions (ConCaps) [36] and BnB-PI in subsequent pretraining steps (see Fig. 3) to reduce the domain gap for downstream VLN tasks.

Previous multi-modal pretraining efforts [24, 28, 16] commonly use two self-supervised losses given image-

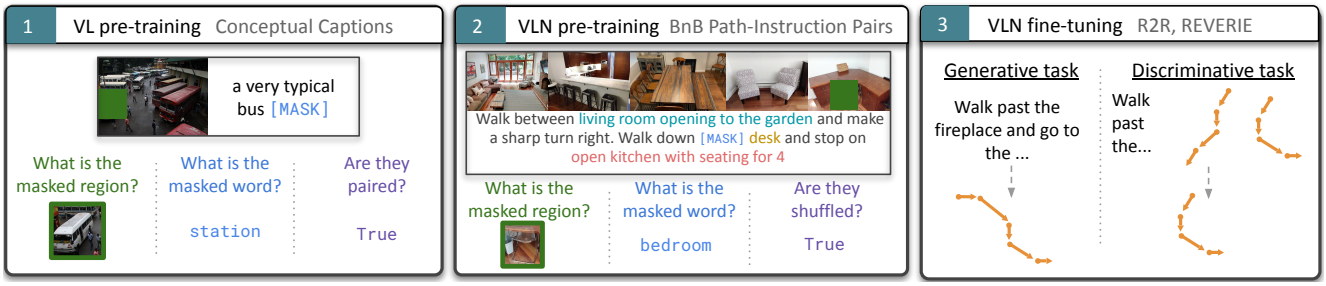


Figure 3: Overview of our pretraining approach. Instead of the usual VL pretraining (panel 1), we adopt in-domain data and use the path-instruction pairs to train Airbert with the masking and shuffling losses (panel 2). We fine-tune Airbert on downstream VLN tasks using both discriminative or generative models (panel 3).

caption (IC) pairs or path-instruction (PI) pairs: (1) *Masking* loss: An input image region or word is randomly replaced by a `[MASK]` token. The output feature of this masked token is trained to predict the region label or the word given its multi-modal context. (2) *Pairing* loss: Given the output features of `[IMG]` and `[CLS]` tokens, a binary classifier is trained to predict whether the image (path) and caption (instruction) are paired.

The above two pretext tasks mainly focus on learning object-word associations instead of reasoning about the temporal order of paths and instructions. For example, if an image V_i appears before V_j, then words from its caption C_i should appear before C_j. In order to promote such a temporal reasoning ability, we propose an additional *shuffling* loss to enforce alignment between PI pairs.

Given an aligned PI pair $X^+ = \{(V_k, C_k)\}_{k=1}^K$, we generate $\mathcal{N}$ negative pairs $X_n^- = \{(V_k, C_l)\}, k \neq l$, by shuffling the composed images or the captions. We train our model to choose the aligned PI pair as compared to the shuffled negatives by minimizing the cross-entropy loss:

$$L = -\log \frac{\exp(f(X^+))}{\exp(f(X^+)) + \sum_n \exp(f(X_n^-))}, \quad (3)$$

where $f(X)$ denotes the similarity score (logit) computed via Airbert for the PI pair X.

4.3. Adaptations for Downstream VLN tasks

We consider two VLN tasks: goal-oriented navigation (R2R [5]) and object-oriented navigation (REVERIE [34]). Airbert can be readily integrated in discriminative and generative models for the above VLN tasks.

Discriminative Model: Navigation as Path-Selection [28]. The navigation problem on the R2R dataset is formulated as a path selection task in [28]. Several candidate paths are generated via beam search from a navigation agent such as [40], and a discriminative model is trained to choose the best path among them. We fine-tune Airbert on the R2R dataset for path selection. A two-stage fine-tuning process is adopted: in the first phase, we use *masking* and *shuffling* losses on the PI pairs of the target VLN dataset in a manner similar to BnB PI pairs; in the second phase, we choose a positive candidate path as one that arrives within 3m of the goal, and contrast it against 3 negative candidate paths. We also compare multiple strategies to mine additional negative pairs (other than the 3 negative candidates), and in fact, empirically show that negatives created using shuffling outperform other options.

Generative Model: Recurrent VLN-BERT [15]. The Recurrent VLN-BERT model adds recurrence to a state in the transformer to sequentially predict actions, achieving state-of-the-art performance on R2R and REVERIE tasks. We use our Airbert architecture as its backbone and apply it to the two tasks as follows. First, the language transformer encodes the instruction via self-attention. Then, the embedded `[CLS]` token in the instruction is used to track history and concatenated with visual tokens (observable navigable views or objects) in each action step. Self-attention and cross-attention on embedded instructions are employed to update the state and visual tokens and the attention score from the state token to visual tokens is used to decide the action at each step. We fine-tune the Recurrent VLN-BERT model with Airbert as the backbone in the same way as [15].

Additional details about the models and their implementation are provided in the Appendix B.

5. Experimental Results

We first perform ablation studies evaluating alternative ways to pretrain Airbert in Sec. 5.1. Then, we compare Airbert with state-of-the-art methods on R2R and REVERIE tasks in Sec. 5.2. Finally, in Sec. 5.3, we evaluate models in a more challenging setup: VLN few-shot learning where an agent is trained on examples taken from one/few houses.

R2R Setup. Most of our experiments are conducted on the R2R dataset [5], where we adopt standard splits and metrics defined by the task. We focus on success rate (SR), which is the ratio of predicted paths that stop within 3m of the goal.

	Cat	Instruction		Path		SR on Val	
		Rep	Gen	Merge	Insert	Seen	Unseen
1	-	-	-	-	-	71.21	62.45
2	✓	-	-	-	-	73.84	62.71
3	-	✓	-	-	-	72.67	63.35
4	-	-	✓	-	-	71.19	63.11
5	-	-	-	✓	-	70.51	64.07
6	-	-	-	-	✓	74.43	66.05
7	-	✓	-	✓	✓	73.57	**66.52**

Table 2: Comparison between various BnB PI pair creation strategies for pretraining. The first row denotes the use of image-caption pairs. All methods from the second row use masking and shuffling during pretraining. Cat: naive concatenation; Rep: instruction rephrasing; Gen: instruction generation; Merge: image merging; and Insert: captionless image insertion.

	BnB		Speaker		R2R		SR on Val	
	Mask	Shuf.	Rank	Shuf.	Rank	Shuf.	Seen	Unseen
1	-	-	-	-	✓	-	70.20	59.26
2	-	-	✓	✓	✓	✓	73.12	65.50
3	✓	-	-	-	✓	-	73.24	64.21
4	✓	✓	-	-	✓	-	73.57	66.52
5	✓	✓	-	-	✓	✓	74.69	66.90
6	✓	-	✓	-	✓	-	70.21	65.52
7	✓	✓	✓	✓	✓	✓	73.83	**68.67**

Table 3: Impact of shuffling during pretraining and fine-tuning. While additional data helps, we see that using the shuffling loss (abbreviated as Shuf.) consistently improves model performance. Row 1 corresponds to VLN-BERT [28].

	Fine-tuning Strategies	Additional Negatives	SR on Val Seen	SR on Val Unseen
1	VLN-BERT [28]	0	70.20	59.26
2	(1) + Wrong trajectories	2	70.11	59.11
3	(1) + Highlight keywords	0	71.89	61.37
4	(1) + Hard negatives	2	71.89	61.63
5	(1) + Shuffling (Ours)	2	72.46	**61.98**

Table 4: Comparison between different strategies for fine-tuning a ViLBERT model on the R2R task. VLN-BERT [28] fine-tunes ViLBERT with a masking and ranking loss. Each row (described in the text) is an independent data augmentation and can be compared directly against the baseline (row 1).

Please refer to [5, 28] for a more detailed explanation of the metrics. In particular, as the discriminative model uses path selection for R2R, we follow the pre-explored environment setting adopted by VLN-BERT [28].

REVERIE Setup. We also adopt standard splits and metrics on the REVERIE task [34]. Here, the success rate (SR) is the ratio of paths for which the agent stops at a viewpoint where the target object is visible. Remote Grounding Success Rate (RGS) measures accuracy of localizing the target object in the stopped viewpoint, and RGS per path length (RGSPL) is a path length weighted version.

Model	Replace-Nouns		Swap-Nouns		Directions	
	Seen	Unseen	Seen	Unseen	Seen	Unseen
VLN-BERT	60.3	58.7	53.4	52.3	46.2	45.3
Airbert	68.3	66.6	66.6	61.1	47.3	49.8

Table 5: Accuracy of models attempting to pick the correct PI pair from a pool of correct + 10 negatives created by simple corruptions such as replacing or swapping noun phrases and switching directions (left with right). Random performance is $\frac{1}{11}$ or 9.1%.

5.1. Pretraining with BnB

We perform ablation studies on the impact of various methods for creating path-instruction pairs. We also present ablation studies that highlight the impact of using the shuffling loss during Airbert's pretraining as well as fine-tuning stages. Throughout this section, our primary focus is on the SR on the unseen validation set and we compare our results against VLN-BERT [28], which achieves a SR of 59.26%.

1. Impact of creating path-instruction pairs. Table 2 presents the performance of multiple ways of using the BnB dataset after ConCaps pretraining as illustrated in Fig. 3. In row 1, we show that directly using BnB IC pairs without any strategies to reduce domain gap improves performance over VLN-BERT by 3.2%. Even if we skip ConCaps pretraining, we achieve 60.54% outperforming 59.26% of VLN-BERT. It proves that our BnB dataset is more beneficial to VLN than the generic ConCaps dataset.

Naive concatenation (row 2) does only slightly better than using the IC pairs (row 1) as there are still domain shifts with respect to fluency of transitions and lack of visual context. Rows 3-6 show that each method mitigates domain-shift to some extent. Instruction rephrasing (row 3) performs better at improving instructions than instruction generation (row 4), possibly since the generator is unable to use the diverse vocabulary of the BnB captions. Inserting captionless images at random locations (row 6) reduces the domain-shift significantly and achieves the highest individual performance. Finally, a combination of instruction rephrasing, image merging and captionless insertion provides an overall 3.8% improvement over concatenation, and a large 7.2% improvement over VLN-BERT.

2. Shuffling loss applied during pretraining. Table 3 demonstrates that shuffling is an effective strategy to train the model to reason about temporal order, and enforce alignment between PI pairs. Rows 3-5 show that shuffling is beneficial both during pretraining with BnB-PI data, or during fine-tuning with R2R data, and results in 2.3% and 0.4% improvements respectively. In combination with the *Speaker* dataset (paths from seen houses with generated instruction yielding 178K additional PI pairs [40]), we see that the shuffling loss provides 3.1% overall improvement (row 6

	Airbert	VLN-BERT [28]	Speaker [40]	Follower [40]	Val Seen					Val Unseen				
					PL	NE	SPL	OSR	SR	PL	NE	SPL	OSR	SR
1	-	✓	-	-	10.28	3.73	0.66	76.47	70.20	9.60	4.10	0.55	69.22	59.26
2	✓	-	-	-	10.59	3.21	0.69	80.71	**73.85**	10.03	3.24	0.63	78.45	**68.67**
3	-	-	✓	✓	10.69	2.72	0.70	82.94	74.22	10.10	3.32	0.63	76.63	67.90
4	-	✓	✓	✓	10.61	2.35	0.78	86.57	**81.86**	10.00	2.76	0.68	81.91	73.61
5	✓	-	✓	✓	10.63	2.13	0.77	87.17	**81.40**	9.99	2.69	0.70	82.89	**75.01**

Table 6: Performance of single models and the impact of ensembling VLN-BERT or Airbert with the speaker and follower.

Methods	Validation Seen						Validation Unseen						Test Unseen					
	Navigation				RGS	RGSPL	Navigation				RGS	RGSPL	Navigation				RGS	RGSPL
	SR	OSR	SPL	TL			SR	OSR	SPL	TL			SR	OSR	SPL	TL		
Seq2Seq-SF [5]	29.59	35.70	24.01	12.88	18.97	14.96	4.20	8.07	2.84	11.07	2.16	1.63	3.99	6.88	3.09	10.89	2.00	1.58
RCM [47]	23.33	29.44	21.82	10.70	16.23	15.36	9.29	14.23	6.97	11.98	4.89	3.89	7.84	11.68	6.67	10.60	3.67	3.14
SMNA [26]	41.25	43.29	39.61	7.54	30.07	28.98	8.15	11.28	6.44	9.07	4.54	3.61	5.80	8.39	4.53	9.23	3.10	2.39
FAST-MATTN [34]	**50.53**	**55.17**	**45.50**	**16.35**	31.97	29.66	14.40	28.20	7.19	45.28	7.84	4.67	19.88	30.63	11.61	39.05	11.28	6.08
Rec (OSCAR) [15]	39.85	41.32	35.86	12.85	24.46	22.28	25.53	27.66	21.06	14.35	14.20	12.00	24.62	26.67	19.48	14.88	12.65	10.00
Rec (ViLBERT)	43.64	45.61	37.86	15.75	31.69	27.58	24.57	29.91	19.81	17.83	15.14	12.15	22.17	25.51	17.28	18.22	12.87	10.00
Rec (VLN-BERT)	41.11	42.87	35.55	15.62	28.39	24.99	25.53	29.42	20.51	16.94	16.42	13.29	23.57	26.83	18.73	17.63	14.24	11.63
Rec (Airbert)	47.01	48.98	42.34	15.16	**32.75**	**30.01**	**27.89**	**34.51**	**21.88**	**18.71**	**18.23**	**14.18**	**30.28**	**34.20**	**23.61**	**17.91**	**16.83**	**13.28**

Table 7: Navigation and object localization performance on the REVERIE dataset, including results on the unseen test set (leaderboard).

vs. 7). The BnB-PI data brings more improvements than the Speaker dataset (row 2 vs. 5). Putting together the BnB-PI data, Speaker dataset and shuffling, we achieve 68.67% SR on the R2R dataset with a single model.

3. Shuffling loss applied during fine-tuning. The final stage of model training on R2R involves fine-tuning to rank multiple candidate paths that form the path selection task. We compare various approaches to improve this fine-tuning procedure (results in Table 4). (1) In row 2, we explore the impact of using additional negative paths. Unsurprisingly, this does not improve performance. (2) Inspired by [12], we highlight keywords in the instruction using a part-of-speech tagger [17], and include an extra loss term that encourages the model to pay attention to their similarity scores (row 3). (3) Another alternative suggested by [12] involves masking keywords in the instruction and using VLP models to suggest replacements, resulting in hard negatives (row 4).

Hard negatives and highlighting keywords improve performance by 2.1-2.3%, but at the cost of extra parsers or VLP models. In contrast, shuffling visual paths to create two additional negatives results in highest improvement (row 5, +2.7% on val unseen) and appears to be a strong strategy to enforce temporal order reasoning, that neither requires external parsers nor additional VLP models.

4. Error analysis. We study the areas in which Airbert brings major improvements by analyzing scores for aligned PI pairs and simple corruptions that involve replacing noun phrases (*e.g. bedroom* by *sofa*), swapping noun phrases appearing within the instruction, or switching left and right directions (*e.g. turn left/right* or *leftmost/rightmost chair*). In particular, for every ground-truth aligned PI pair, we create 10 additional negatives by corrupting the instruction, and measure the accuracy of the model selecting the correct pair. Table 5 shows that Airbert with in-domain training and the shuffling loss achieves large improvements ($> 8\%$) for corruptions involving replacement or swapping of noun phrases. On the other hand, distinguishing directions continues to be a challenging problem; but here as well we see Airbert outperform VLN-BERT by 4.5%.

Model	Test Unseen				
	PL	NE	SPL	OSR	SR
Speaker-Follower [11]	1,257	4.87	0.01	96	53
PreSS [22]	10.5	24.5	0.63	57	53
PREVALENT [13]	10.21	4.52	0.56	64	59
Self-Monitoring [26]	373	4.48	0.02	97	61
Reinforced CM [47]	358	4.03	0.02	96	63
EnvDrop [5]	687	3.26	0.01	99	69
AuxRN [51]	41	3.24	0.21	81	71
VLN-BERT [28]	687	3.09	0.01	99	73
Airbert (ours)	687	2.69	0.01	99	77

Table 8: Navigation performance on the R2R unseen test set as indicated on the benchmark leaderboard.

5.2. Comparison against state-of-the-art

R2R. We first evaluate the discriminative model for the R2R task. Similar to VLN-BERT, we evaluate Airbert as an ensemble model created by a linear combination (chosen through grid search) of multiple model outputs (see Table 6). First, we see that Airbert alone (row 2) outperforms VLN-BERT (row 1) by 9.4% on the unseen environments

# Env.	Traj.	VLN-BERT SR		Airbert SR	
		Seen	Unseen	Seen	Unseen
1	Rand	45.71	22.43	47.88	50.00
6	Rand	52.75	35.99	54.48	57.97
61	Rand	67.68	57.15	64.24	65.60
61	[40]	70.20	59.26	73.83	68.48

Table 9: Performance on R2R few-shot evaluation. During training, only a subset of the Matterport [7] environments are accessible. Standard deviation is reported in the supplementary material.

and a strong ensemble of speaker and follower models [40] (row 3) by 0.7%. Ensembling Airbert results in a gain of 1.4% over the VLN-BERT ensemble (row 4 vs. 5).

We also obtain results on the test set by submitting our best method to the R2R leaderboard[5]. As seen from Table 8, our method of ensembling Airbert, speaker, and follower (similar to VLN-BERT with speaker and follower [10]) achieves the highest success rate at 77% and is ranked first as of the submission deadline. Both VLN-BERT and Airbert use 30 candidate trajectories sampled by beam search with EnvDrop [40], inducing the same path length (PL) for the three methods. As the SPL metric on the leaderboard takes into account the total path length over the 30 trajectories, the SPL is very low and similar across the approaches. Airbert also benefits generative models for the R2R task. The results are presented in the Appendix C.

REVERIE. Table 7 presents results for the REVERIE dataset. The last four rows in the table use Recurrent VLN-BERT [15] with different backbones or parameter initialization. The OSCAR and ViLBERT backbones are pretrained on out-of-domain image-caption pairs. As compared to OSCAR, we observe slight improvements using the ViLBERT backbone for the REVERIE task. VLN-BERT shares the same architecture as ViLBERT, but is pretrained on the R2R dataset, resulting in performance improvement on the unseen environments. Our pretrained Airbert achieves significantly better performance than VLN-BERT, with over 2.4% gain on navigation SR and 1.8% gain on RGS in unseen environments (val unseen). Without any special adaptation, we see that Airbert brings benefits from pretraining on the BnB dataset. We also achieve the state-of-the-art performance on the REVERIE test set by the time of submission, surpassing previous works by a large margin.

5.3. Training a navigation agent on few houses

We hypothesize that in-domain pretraining, especially one that leverages proposed PI pair generation methods, can achieve superior performance while requiring less training data. To evaluate this, we propose a novel few shot evaluation paradigm for VLN: models are allowed to fine-tune on samples (PI pairs) from one (or few) environments. Few-shot learning for VLN is particularly interesting as visual appearance of houses may differ vastly across geographies, and while training data is hard to obtain, pretraining data like BnB may be readily available.

One/few shot tasks. We considered two types of setups: (1) learning from a single environment, which we refer as one-shot learning; and (2) learning from 6 environments (representing 10% of the total training size). For both cases, we randomly sample 5 sets of environments, and report average results (standard deviations in the Appendix C). As the number of paths in an environment may have a large impact on performance, we exclude 17 of 61 environments with less than 80 paths.

Results. We adopt VLN-BERT, pretrained on ConCaps, as a baseline for few-shot tasks. Recall that fine-tuning VLN-BERT and Airbert on R2R relies on candidate paths drawn from an existing model (EnvDrop [40]). However, as this would lead to unfair comparisons (EnvDrop is trained on the full dataset), candidate paths are sampled as the shortest path between two random positions.

Table 9 shows that Airbert largely outperforms VLN-BERT on the unseen validation set: 27.6% with 1 house and 22% with 6 houses. Airbert fine-tuned on 6 houses is almost as good as VLN-BERT on the entire training set. The last two rows of the table shows that using random paths does not lead to a large performance drop for both models and is a testament to the power of pretrained networks.

6. Conclusion

We introduced BnB, a large-scale, in-domain, image-text dataset from houses listed on online rental marketplaces and showed how domain gaps between BnB image-caption pairs and VLN tasks can be mitigated through the creation of path-instruction pairs. We also proposed shuffling, as a means to improve an agent's reasoning about temporal order. Our pretrained model Airbert, achieved state-of-the-art on R2R through the discriminative path-selection setting, and REVERIE through a generative setting. We also demonstrated large performance improvements when applying our model to a challenging one/few-shot VLN setup, highlighting the impact of good pretraining in VLN tasks.

Acknowledgments. This work was granted access to the HPC resources of IDRIS under the allocation 101002 made by GENCI. It was funded in part by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR19-P3IA-0001 (PRAIRIE 3IA Institute) and by Louis Vuitton ENS Chair on Artificial Intelligence.

[5] https://eval.ai/web/challenges/challenge-page/97/overview also shows performance for ensembling Airbert, VLN-BERT, speaker and follower at a unseen test set SR of 78%.

References

[1] Airbnb fast facts. Accessed: 2021-03-13 at https://news.airbnb.com/fast-facts/. 3

[2] Chris Alberti, Jeffrey Ling, Michael Collins, and David Reitter. Fusion of detected objects in text for visual question answering. In *EMNLP*, 2019. 1

[3] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018. 1, 2

[4] 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 *CVPR*, 2018. 4, 13, 14

[5] 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 *CVPR*, 2018. 1, 2, 3, 4, 5, 6, 7, 15

[6] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. VQA: Visual Question Answering. In *ICCV*, 2015. 2

[7] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *3DV*, 2017. 3, 8, 17

[8] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *CVPR*, 2019. 2

[9] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020. 1, 2

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 1, 2, 4, 8

[11] 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 *NIPS*, 2018. 2, 7, 15

[12] Tanmay Gupta, Arash Vahdat, Gal Chechik, Xiaodong Yang, Jan Kautz, and Derek Hoiem. Contrastive learning for weakly supervised phrase grounding. In *ECCV*, 2020. 7

[13] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *CVPR*, 2020. 1, 2, 7, 14, 15

[14] Yicong Hong, Cristian Rodriguez, Qi Wu, and Stephen Gould. Sub-instruction aware vision-and-language navigation. In *EMNLP*, 2020. 3

[15] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. A recurrent vision-and-language BERT for navigation. *arXiv preprint arXiv:2011.13922*, 2021. 2, 5, 7, 8, 14, 15

[16] Haoshuo Huang, Vihan Jain, Harsh Mehta, Alexander Ku, Gabriel Magalhaes, Jason Baldridge, and Eugene Ie. Transferable representation learning in vision-and-language navigation. In *ICCV*, 2019. 1, 2, 4

[17] V. Joshi, Matthew E. Peters, and Mark Hopkins. Extending a parser to distant domains using a few dozen partially annotated examples. In *ACL*, 2018. 7

[18] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. ReferItGame: Referring to objects in photographs of natural scenes. In *EMNLP*, 2014. 2

[19] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *ECCV*, 2020. 2

[20] 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. *IJCV*, 123:32–73, 2017. 4

[21] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-Across-Room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *EMNLP*, 2020. 2

[22] Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Celikyilmaz, Jianfeng Gao, Noah Smith, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. *EMNLP*, 2019. 1, 7, 15

[23] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *ECCV*, 2020. 1, 2, 14

[24] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NIPS*, 2019. 1, 2, 4

[25] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-task vision and language representation learning. In *CVPR*, 2020. 1

[26] 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 *ICLR*, 2019. 2, 7

[27] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The Regretful Agent: Heuristic-aided navigation through progress estimation. In *CVPR*, 2019. 2

[28] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *ECCV*, 2020. 1, 2, 4, 5, 6, 7, 13, 14

[29] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100M: Learning a text-video embedding by watching hundred million narrated video clips. In *ICCV*, 2019. 2

[30] Khanh Nguyen and Hal Daumé III. Help, Anna! Visual navigation with natural multimodal assistance via retrospective curiosity-encouraging imitation learning. In *ACL*, 2019. 2

[31] Khanh Nguyen, Debadeepta Dey, Chris Brockett, and Bill Dolan. Vision-based navigation with language-based assistance via imitation learning with indirect intervention. In *CVPR*, 2019. 2

[32] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2Text: Describing images using 1 million captioned photographs. In *NIPS*, 2011. 1, 2

[33] Yuankai Qi, Zizheng Pan, Shengping Zhang, Anton van den Hengel, and Qi Wu. Object-and-action aware model for visual language navigation. In *ECCV*, 2020. 2

[34] Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. REVERIE: Remote embodied visual referring expression in real indoor environments. In *CVPR*, 2020. 2, 3, 5, 6, 7

[35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021. 2

[36] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual Captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018. 1, 2, 3, 4

[37] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *CVPR*, 2020. 2

[38] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. In *ICLR*, 2019. 1

[39] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP*, 2019. 2, 14

[40] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*, 2019. 2, 3, 5, 6, 7, 8, 15, 17

[41] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *CoRL*, 2020. 2

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NIPS*, 2017. 1, 2

[43] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. *arXiv preprint arXiv:1506.03134*, 2015. 12

[44] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: Lessons learned from the 2015 MSCOCO image captioning challenge. *PAMI*, 39:652–663, 2016. 2

[45] Hu Wang, Qi Wu, and Chunhua Shen. Soft expert reward learning for vision-and-language navigation. In *ECCV*, 2020. 2

[46] Liwei Wang, Yin Li, and Svetlana Lazebnik. Learning deep structure-preserving image-text embeddings. In *CVPR*, 2016. 2

[47] 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 *CVPR*, 2019. 2, 7

[48] 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 *ECCV*, 2018. 2

[49] Yubo Zhang, Hao Tan, and Mohit Bansal. Diagnosing the environment bias in vision-and-language navigation. In *IJCAI*, 2020. 1

[50] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *PAMI*, 40:1452–1464, 2017. 3, 4, 11, 14

[51] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *CVPR*, 2020. 7

Appendix

In this supplementary material, we present additional details, statistics and examples for the BnB dataset; we discuss implementation details for the models used in our work; and present qualitative results as well as the detailed results for the new few-shot learning paradigm.

A. BnB dataset

This section presents additional details for our *Bed-and-Breakfast* (BnB) dataset. We start by a short discussion of image-caption pairs (BnB IC) collected from an online rental marketplaces and their statistics. Subsequently, we present how a combinatorially large number of path-instruction pairs (BnB PI) can be created automatically. We end this section with multiple examples of BnB PI pairs generated via the concatenation and domain-shift reduction (*e.g.* rephrasing, captionless insertion) strategies.

A.1. Filtering image-caption pairs: Outdoor images

Images of outdoor scenes are almost never seen in the environments used in downstream VLN tasks. In fact, not only are the images out-of-domain (such images are rarely seen in the VLN environments), their captions are often irrelevant to a VLN task. In order to alleviate the impact of such noisy images and captions, we discard outdoor images from the pretraining process. Figure 4 illustrates several examples of misleading outdoor image-caption pairs. Captions as written by the host are presented in the label below the image. The caption for the image in Figure 4a refers to a "*bedroom*", however, the image does not show a bedroom. Similarly, the image-caption pair in the Figure 4b talks about activities or festivals that take place in the neighborhood of the listing, however, they are not relevant for solving indoor navigation tasks. Finally, Figure 4c shows an outdoor scene with several birds along with a noisy caption that is not directly related to the image content, but the emotion that the image may evoke.

A.2. Dataset details and Statistics

BnB image-caption pairs. We collect BnB IC pairs from 150K listings on *Airbnb* resulting in 713K image-caption pairs and 676K images without captions. In Figure 5, we present some key statistics about this data. Figure 5a presents a histogram of the number of images found in each listing. While most listings have less than 20 images, this is still a sufficiently large and diverse in-domain distribution. In Figure 5b, we summarize the rooms depicted in the images through predicted category labels obtained using a CNN trained on the *Places365* dataset [50]. These category labels are used as part of our proposed extensions such as *image merging*.

Creating BnB path-instruction pretraining samples. We create the BnB PI pairs on-the-fly during training to mimic the agent's visual trajectory and a corresponding instruction through an environment. Each sample in a batch is created by randomly sampling a listing without replacement during an epoch (one epoch consists of one PI pair from each listing). Then, the number of IC pairs K that form the PI pair are chosen (as an integer) from a uniform distribution, $K \sim U[4, 7]$. We sample $N \sim U[2, K]$ IC pairs that have a non-empty caption and the remainder $K - N$ images are chosen from the set of captionless images. Any image in the path may include additional visual context (from the same room) via the *image merging* strategy. Similarly, the *instruction rephrasing* strategy may be employed by using existing R2R instruction templates and filling them with noun phrases extracted from the image captions.

The above procedure results in creating one correctly aligned (positive) PI pair, (X^+ in the main paper). To employ the shuffling loss for each sample, we create 9 additional negatives (X_n^- in the paper) by shuffling either the sequence of images or captions, ensuring that the post-shuffling order does not align with the positive pair.

Statistics for BnB PI pairs. Due to the large number of possible combinations, we can (theoretically) create 200 billion path-instruction pairs, using the simple concatenation strategy. This number grows to over 300 quadrillion when considering additional visual context augmentations and fluent instructions.

For *instruction rephrasing*, we create 11,626 fill-in-the-blank templates from the R2R training set. Figure 5c shows the distribution of the number of blanks in the templates – most instruction templates have 2-7 blanks into which we insert noun phrases from the BnB captions.

While we are unable to generate the entire BnB PI dataset for computing statistics, we generate 50K PI pairs as a representative sample. Figure 5d presents the distribution of instruction lengths (number of words) for different datasets. We see that the captions in BnB IC pairs are much shorter than typical instructions in R2R and REVERIE, while our automatically created instructions in BnB PI pairs exhibit a high level of similarity in terms of their length.

A.3. Examples of BnB PI Pairs

Figure 6 presents generated BnB PI pairs using various strategies proposed in our work, including naive concatenation, instruction rephrasing, instruction generation, image merging and captionless image insertion.

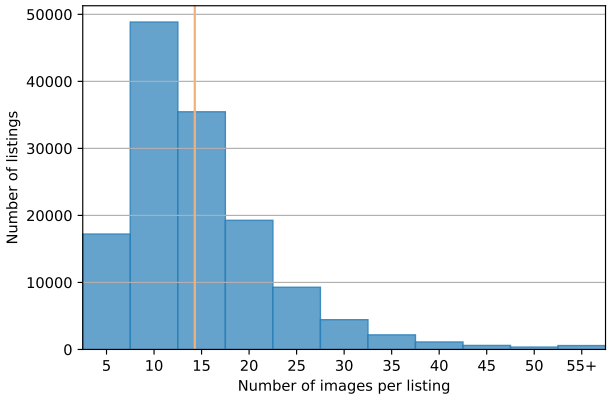
Among the methods to create an instruction, simple concatenation lacks action verbs between sentences for fluent transition leading to a domain shift from real instructions. Instruction rephrasing selects noun phrases from BnB image descriptions and inserts them into real instruction templates, providing a natural feel to the created instruction.

(a) Gorgeous ocean views from bedroom.

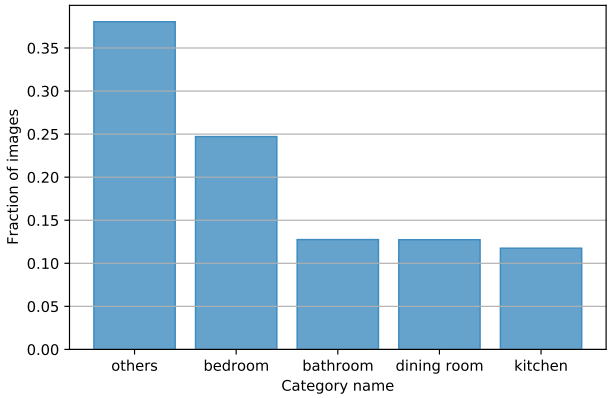
(b) Main stage

(c) Spring excitement

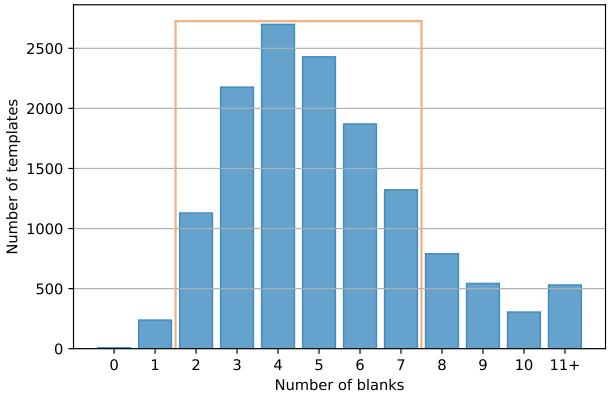
Figure 4: Examples of outdoor images with their corresponding captions.



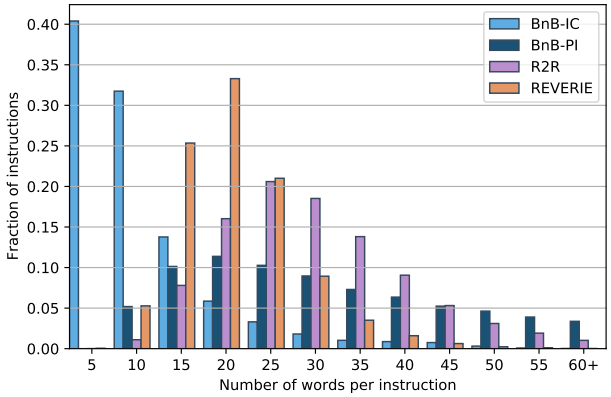
(a) Distribution of the number of images per listing.



(b) Distribution of predicted scene categories on BnB images.



(c) Fill-in-the-blanks templates built using the R2R training set.



(d) Distribution of the instruction lengths.

Figure 5: Statistics of BnB Dataset.

Finally, while the learning approach of instruction generation (recall, this is learned on downstream VLN dataset) produces fluent sentences, it is unable to leverage the diverse captions of BnB images due to the limited vocabulary stemming from the downstream VLN dataset. For example, the generated instruction in Figure 6c does not contain noun phrases related to images in the path. Better caption generation models such as Pointer network [43] may help avoid such problems, however are left for future work.

Among augmentations for path generation, we can see that *image merging* helps to expand relevant visual context from single images to semi-panoramic views, see the bedroom in Figure 6a or the kitchen in Figure 6b. *Captionless image insertion* also improves the path diversity by mimicking unmentioned viewpoints in the instruction (indicated by images with a dotted border).

Concatenation: extra guest room with comfy full bed on top floor of house and top floor shared bathroom for both guest rooms then adjoining modern private bath with stall shower bath and beach towels provided then granny's treasures add a homey touch
Instruction rephrasing: exit extra guest room and turn left. pass top floor shared bathroom then turn right. walk toward a homey touch and wait there.
Instruction generation: walk to the other side of the bathroom and stop next to the last corner on the wall with the candles.

(a) Example 1

Concatenation: full bath and open floor plan living opens to deck, kitchen / dining area
Instruction rephrasing: go around full bath, then open floor plan living down to kitchen / dining area.
Instruction generation: walk into the bathroom and turn right. walk to the end of the landing and turn left. walk into the sitting area and turn right. walk past the chair and stop.

(b) Example 2

Concatenation: bedroom 3 (picture 2 of 2) - 2 twin beds w / full size washer then bedroom 2 - queen bed - 1st floor, bathroom 1st floor. w / tub and shower.
Instruction rephrasing: exit the bedroom 3 and go right into bedroom 2 next to tub and shower.
Instruction generation: walk straight through the doorway and turn right. walk straight through the doorway and turn left. walk through the doorway and stop.

(c) Example 3

Figure 6: Examples of path-instruction pairs created by different strategies. The images with dotted borders are images chosen from the *captionless image insertion* strategy, and the clustered images are from the *image merging* strategy.

B. Implementation details

We present the implementation details for learning Airbert via pretraining using BnB, and subsequent fine-tuning in both discriminative or generative settings.

B.1. Airbert Pretraining

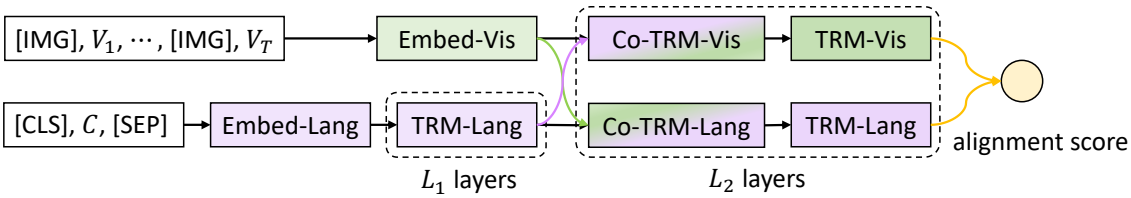
Airbert's architecture is the same as VLN-BERT (see Figure 7a where the number of layers $L_1 = L_2 = 6$). The feature vector v_i^k (corresponding to ith image region of the kth image) is composed of three terms: the first term is the visual feature extracted by the Bottom-Up Top-Down attention model [4]; the second term encodes the location of the region in the image as MLP(l_i^k), where l_i^k is the 5-dim location vector of the given image region defined as the top corner (x, y), the width, height and area; and the last term Emb(k) encodes the position, where Emb is an embedding layer for the image order.

We use 8 V100 SXM2 GPUs (32 GB each) for pretraining Airbert. The model is trained for 15 epochs with a batch size of 64 and learning rate of 4×10^{-5}. Each epoch consists of one randomly sampled PI pair from 95% of the listings, while the remaining 5% are used for validation and preventing overfitting.

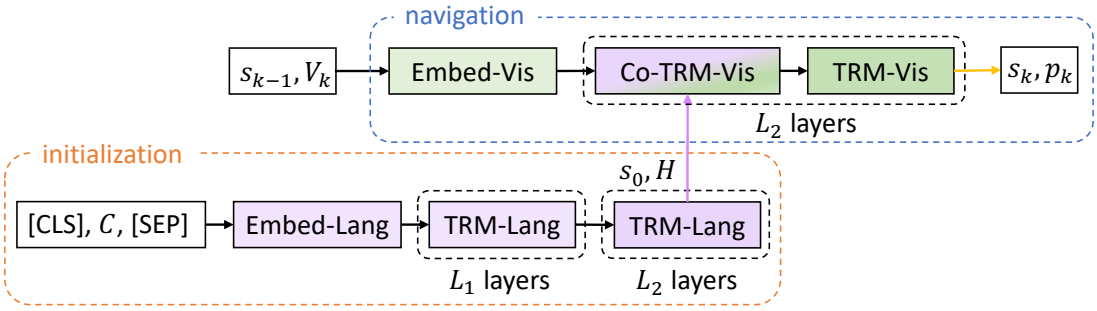
B.2. Fine-tuning in Discriminative Setting

In the discriminative setting, R2R navigation is formulated as a path selection problem given the instruction. The pretrained Airbert model can be directly fine-tuned without any modifications to the architecture to predict the path-instruction alignment (or compatibility) score as shown in Figure 7a.

We follow the same fine-tuning setup as VLN-BERT [28] to allow for a fair comparison. We use the Adam optimizer with a learning rate of 4×10^{-5}. The optimizer is controlled by a learning rate scheduler with a linear warmup and cooldown. We fine-tune Airbert for 30 epochs with a batch size of 64. Samples from the R2R training set are used



(a) Adapting Airbert to a discriminative setting to predict path-instruction alignment score, similar to [28].



(b) Adapting Airbert to a generative setting based on the Recurrent VLN-BERT [15].

Figure 7: The adapted Airbert model in both discriminative and generative settings for downstream VLN tasks.

for fine-tuning and the model checkpoint with the highest success rate on the unseen validation set (val unseen) is selected for the test set and leaderboard submission.

B.3. Fine-tuning in Generative Setting

In the generative setting, an agent is required to predict navigable actions step by step. We adopt the state-of-the-art generative model Recurrent VLN-BERT [15] for R2R and REVERIE tasks. The model uses a pretrained multimodal transformer as a backbone and adds recurrence to a state token to keep track of history for sequential action prediction. Although the original Recurrent VLN-BERT model only implements an LXMERT-like [39] architecture PREVALENT [13], and one-stream BERT-like architecture OSCAR [23], it is easy to plug our two-stream ViLBERT architecture as the backbone.

The adapted model is shown in Figure 7b. For initialization, the language stream is used to encode the instruction C into an instruction representation H. As no visual inputs are used during the initialization, the co-attention modules in the original language stream of ViLBERT are removed, and the output feature of the `[CLS]` token is used as the agent's initial state s_0. For navigation at each step k, the visual stream takes the previous state s_{k-1}, visual observations V_k at step k and the encoded language features H to generate a new state s_k and action decision p_k.

When fine-tuning on the R2R dataset, we use scene features with a ResNet-152 pretrained on Places365 [50] and augment the training data with generated path-instruction pairs from [13]. We train the model via imitation learning and A2C reinforcement learning for 300,000 iterations with a batch size of 16 and learning rate of 10^{-5}. When fine-tuning on the REVERIE dataset, object features encoded by a Bottom-Up Top-Down attention model [4] are used along with the scene features. The model is trained for 200,000 iterations with a batch size of 8. All the experimental setups for fine-tuning are the same as [15] for a fair comparison.

C. Results

In this section, we present additional results on adapting Airbert to a generative setting and applying it to the R2R task. Through several qualitative examples, we obtain a better understanding for Airbert's performance improvements, and finally present detailed results on the new few-shot learning paradigm in VLN.

C.1. Results on R2R with Generative Models

Table 10 shows the performance of different generative models on the R2R dataset. The OSCAR and ViLBERT backbones for Recurrent VLN-BERT [15] (Rec) are all pretrained on large-scale out-of-domain image-caption pairs with object features and similar self-supervised tasks. On the other hand, the PREVALENT [13] backbone is pretrained on in-domain R2R dataset with scene features and fine-tuned with an additional action prediction task. We suspect that this is the reason for PREVALENT's higher performance as compared to using OSCAR or VLN-BERT as backbones. Note that our Airbert backbone is not fine-tuned further on downstream tasks after pretraining.

Methods	Validation Seen				Validation Unseen				Test Unseen			
	TL	NE	SR	SPL	TL	NE	SR	SPL	TL	NE	SR	SPL
Seq2Seq-SF [5]	11.33	6.01	39	-	8.39	7.81	22	-	8.13	7.85	20	18
Speaker-Follower [11]	-	3.36	66	-	-	6.62	35	-	14.82	6.62	35	28
PRESS [22]	10.57	4.39	58	55	10.36	5.28	49	45	10.77	5.49	49	45
EnvDrop [40]	11.00	3.99	62	59	10.70	5.22	52	48	11.66	5.23	51	47
PREVALENT [13]	10.32	3.67	69	65	10.19	4.71	58	53	10.51	5.30	54	51
Rec (no init. OSCAR) [15]	9.78	3.92	62	59	10.31	5.10	50	46	11.15	5.45	51	47
Rec (OSCAR) [15]	10.79	3.11	71	67	11.86	4.29	59	53	12.34	4.59	57	53
Rec (PREVALENT) [15]	11.13	2.90	72	68	12.01	3.93	63	57	12.35	4.09	63	57
Rec (ViLBERT)	11.16	2.54	75	71	12.44	4.20	60	54	-	-	-	-
Rec (VLN-BERT)	10.95	3.37	68	64	11.33	4.19	60	55	-	-	-	-
Rec (Airbert)	11.09	2.68	75	70	11.78	4.01	62	56	12.41	4.13	62	57

Table 10: Navigation performance of different generative models on the R2R dataset.

Replacing OSCAR's single BERT-like architecture with the ViLBERT architecture slightly improves the performance (similar to our results on the REVERIE dataset presented in the main paper). The VLN-BERT model further fine-tunes ViLBERT on the R2R dataset (with the masking loss). This is beneficial to the navigation performance on the unseen environments validation set[6]. Our Airbert initialization achieves substantial performance improvement as compared to the OSCAR and VLN-BERT backbones on unseen environments, while achieving comparable performance with the PREVALENT initialization.

C.2. Qualitative results

We visualize the predicted paths from VLN-BERT and Airbert models. In the following figures, ● is the starting viewpoint of the agent, ■ denotes viewpoints in the ground-truth path, ■ for VLN-BERT and ■ for Airbert. Arrows indicate the navigation direction.

New houses. In Figure 8, we compare predicted paths from VLN-BERT and Airbert in new houses beyond the training environments. Benefiting from BnB dataset that provides diverse visual environments in pretraining, our Airbert model generalizes better to recognize different room types in new houses (see Figure 8a-8d), and performs better on significantly different environments such as a church (Figure 8e) or castle (Figure 8f).

New objects. Airbert also improves the understanding of new objects in home environments, *e.g.* through noun phrases related to household objects. As shown in Figure 9, it is successful at following instructions containing noun phrases that rarely occur or are even unseen on the training set, while the VLN-BERT model that is trained on a large image-caption corpus not pertaining to houses fails.

[6]The performance of VLN-BERT on the seen validation set is lower because the model checkpoint is selected to maximize performance on validation unseen set which happens to be at an earlier iteration.

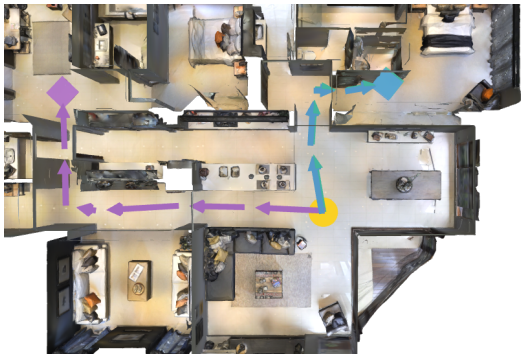
Similar environments and instructions. Figure 10 displays examples where the environments and the instructions are similar to those on the training set, with the aim to show that the shuffling loss in pretraining also benefits learning. For example, in Figure 10a, the VLN-BERT agent ■ focuses on the stairs (in the last step) and goes upstairs incorrectly, whereas Airbert learns to consider intermediate steps such as "*lounge chairs*" and "*cabinet*" besides the last step by learning from the shuffling task. Similarly, in Figure 10c, we see that the VLN-BERT agent stops at the wrong stairs, while Airbert considers intermediate steps such as "*hallway*" and "*wooden doors*", and ends within the acceptable range of 3m from the goal.

Failure cases. Figure 11 presents some failure cases for both VLN-BERT and Airbert. It reveals that current models still struggle to deal with relationships such as "*between*" (Figure 11a), or directional instructions such as "*on the left*" (Figure 11b). Similar failures are also highlighted by Table 5 of the main paper where we show that models fail to choose the correct instruction when a direction keyword (left/right) is switched.

C.3. Few-shot Learning on VLN

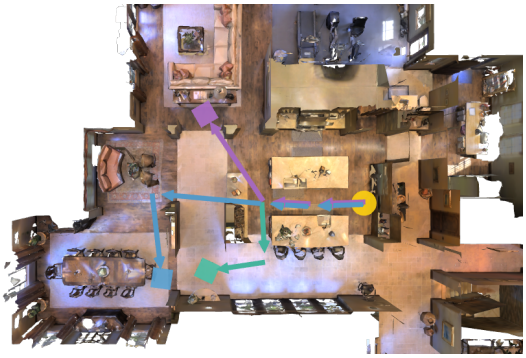
As mentioned in the main paper, we present complete results for the few-shot learning evaluation, along with standard deviations in Table 11. While the performance on the seen validation houses fluctuates a lot (also due to changing the environment in the seen validation set), unseen validation is very stable. Recall that VLN-BERT achieves an unseen validation performance of 27% and 37% with 1 and 6 training environments respectively. On the other hand, Airbert achieves a superior 49.5% and 58% – an absolute improvement of ∼22% in both cases.

(a) **R2R ✔:** Walk over the kitchen counter, turn left, walk ahead till wall, turn right, walk to the closet room, wait at front.

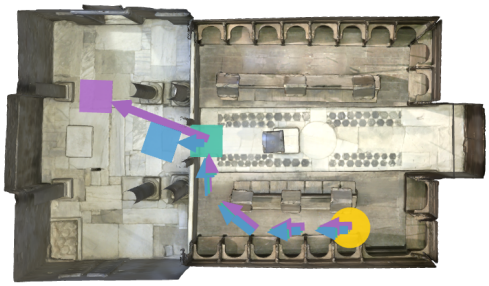


(b) **REVERIE ✔:** Walk past the kitchen and enter the hallway. Turn right at the artwork and wait by the closet.

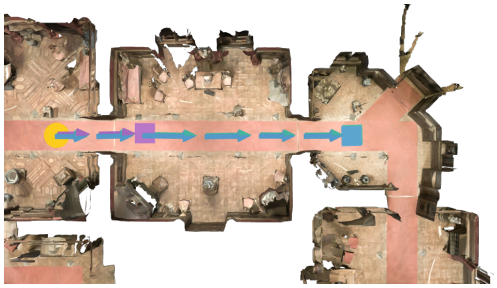
(c) **R2R ✔:** Walk forward to the sitting area to the right of the stairs. Walk to the wall of windows and take a right into the recreation room and stop before you reach the pool table.



(d) **R2R ✔:** Go between the counters, turn left, turn right, and stop before the display and dining room.

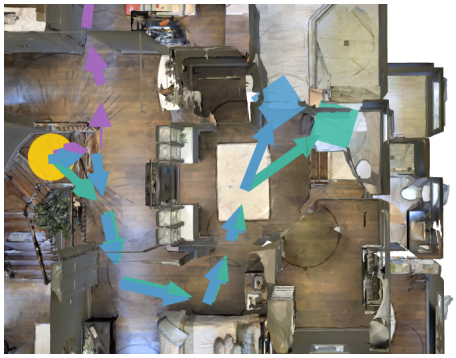


(e) **R2R ✔:** Turn right and head towards the end. Once you reach the end make a right and stop.



(f) **R2R ✔:** Walk straight out the door in front of you and follow the red carpet. Keep going through the room with the ropes and stop when you enter the next room with ropes.

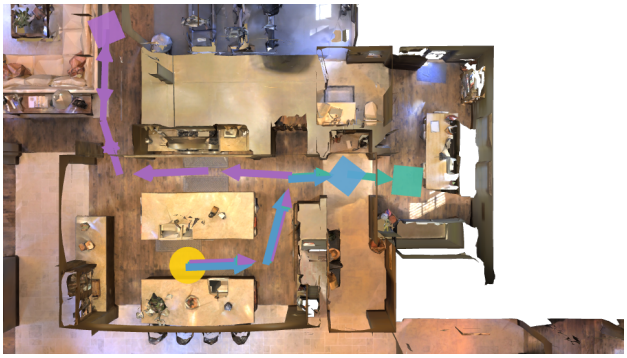
Figure 8: When navigating in new houses, our Airbert model not only successfully recognizes the closet room in (a) and (b), pool table (c), living room (d), but also generalizes better to challenging environments, such as the church (e) and castle (f).



(a) **R2R ✔:** Walk up the stairs and take a right. Walk into the bedroom and take a left . Take another left at the night stand and walk out of the bedroom. Wait by the toilet in the second door on the right.

(b) **R2R ✔:** Go straight past the table and chairs then turn left and continue to go past the table and chairs. Wait near the white antique furniture with the two chairs on on each side.

(c) **REVERIE ✔:** Walk past the pool table and towards the TV on the far side of the room and grab the coffee table that is located in front of the couch



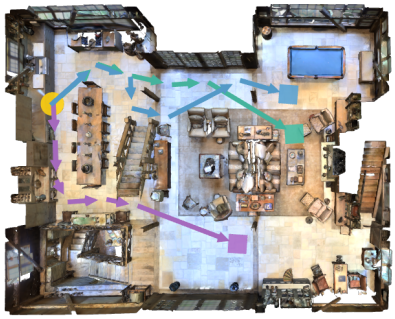
(d) **REVERIE ✔:** Please go to the pantry room with the two large freezers and kitchen appliances on the large table and reset the flipped breaker in the breaker panel box to the right of the freezers

Figure 9: The Airbert model outperforms VLN-BERT to recognize rare or even unseen objects in training set. (a) Rare object "*night stand*"; (b) unseen object "*antique furniture*"; (c) rare object "*pool table*"; and (d) unseen object "*freezer*".

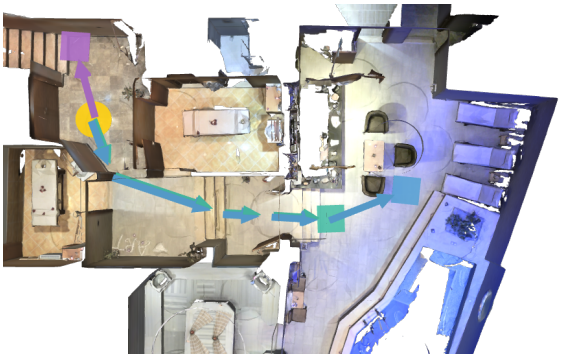
# Env.	Traj.	Val Seen SR					Val Unseen SR				
		PL	NE	SPL	OSR	SR	PL	NE	SPL	OSR	SR
1	Rand	10.97	5.36	0.44	63.74	47.87 ±0.03	10.84	4.86	0.51	68.46	54.48 ±0.04
6	Rand	9.84	5.49	0.47	65.93	50.00 ±0.02	9.55	4.55	0.55	70.89	57.97 ±0.01
61	Rand	10.91	4.87	0.60	76.23	64.24	9.50	3.70	0.62	76.24	65.60
61	[40]	10.59	3.21	0.69	80.71	73.85	10.03	3.24	0.63	78.45	68.67

Table 11: Performance of Airbert on R2R few-shot evaluation. During training, only a subset of the Matterport [7] environments are accessible.

(a) **R2R ✔:** Walk from dining room to living room turning slightly right before lounge chairs, walk straight following cabinet. Turn slight right and stop at stairs.



(b) **R2R ✔:** Walk on into the kitchen and turn to the right. Walk past the staircase, behind the chairs. Walk to the right of the pillar. Stop and wait by the footstool.

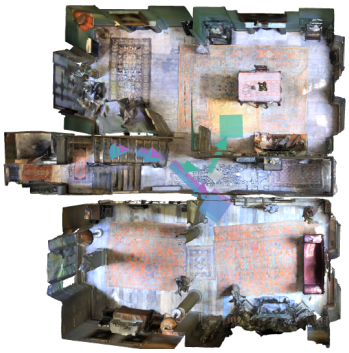


(c) **R2R ✔:** Walk out of the hallway and turn left. Walk down the steps and through the wooden doors. Walk down the steps and stop.

(d) **R2R ✔:** Go straight passed the coffee table turn left and go through the left door to the stairs. Stop in front of the stairs.

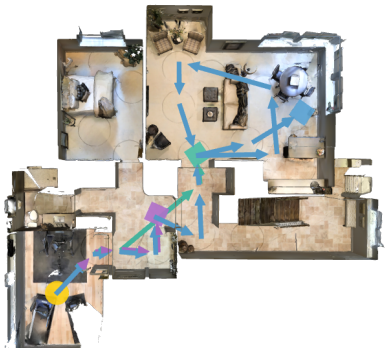
Figure 10: Examples in similar environments and instructions to the training set. The improvements of Airbert model can be contributed to the shuffling loss in pretraining.

(a) **R2R ✘:** Walk between the two kitchen islands and then turn right. Pass through the stone archway and stop just after you pass through it. Wait there.



(b) **R2R ✘:** Exit the bathroom and go down the stairs. Enter the last doorway on the left and stop just before stepping on the rug.

(c) **REVERIE ✘:** go to level 3 bathroom in the first bedroom left of the stairs and grab the mirror on the wall



(d) **REVERIE ✘:** Go to the lounge on this level and polish the black leather armchair in the corner

Figure 11: Failure cases for both VLN-BERT and Airbert models.