Stay on the Path: Instruction Fidelity in Vision-and-Language Navigation

Vihan Jain[*]   **Gabriel Magalhaes**[*]   **Alexander Ku**[*]   **Ashish Vaswani**
Eugene Ie   **Jason Baldridge**
Google Research
{vihan, gamaga, alexku, avaswani, eugeneie, jridge}@google.com

Abstract

Advances in learning and representations have reinvigorated work that connects language to other modalities. A particularly exciting direction is Vision-and-Language Navigation (VLN), in which agents interpret natural language instructions and visual scenes to move through environments and reach goals. Despite recent progress, current research leaves unclear how much of a role language understanding plays in this task, especially because dominant evaluation metrics have focused on goal completion rather than the sequence of actions corresponding to the instructions. Here, we highlight shortcomings of current metrics for the Room-to-Room dataset (Anderson et al., 2018b) and propose a new metric, Coverage weighted by Length Score (CLS). We also show that the existing paths in the dataset are not ideal for evaluating instruction following because they are direct-to-goal shortest paths. We join existing short paths to form more challenging extended paths to create a new data set, Room-for-Room (R4R). Using R4R and CLS, we show that agents that receive rewards for instruction fidelity outperform agents that focus on goal completion.

1 Introduction

In Vision-and-Language Navigation (VLN) tasks, agents must follow natural language navigation instructions through either simulated (Macmahon et al., 2006; Yan et al., 2018; Bisk et al., 2018; Shah et al., 2018), simulations of realistic (Blukis et al., 2018; Misra et al., 2018) and real environments (Anderson et al., 2018b; de Vries et al., 2018; Chen et al., 2019; Cirik et al., 2018), or actual physical environments (Skočaj et al., 2016; Thomason et al., 2018; Williams et al., 2018). Compared to other tasks involving co-grounding in visual and
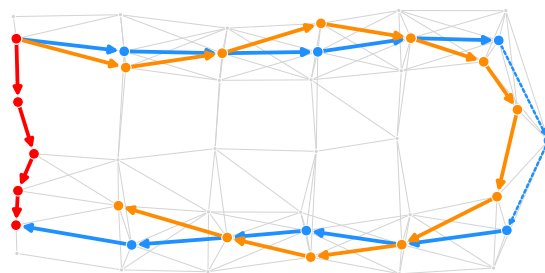

Figure 1: *It's the **journey**, not just the goal.* To give language its due place in VLN, we compose paths in the R2R dataset to create longer, twistier R4R paths (blue). Under standard metrics, agents that head straight to the goal (red) are not penalized for ignoring the language instructions: for instance, SPL yields a perfect 1.0 score for the red and only 0.17 for the orange path. In contrast, our proposed CLS metric measures fidelity to the reference path, strongly preferring the agent with the orange path (0.87) over the red one (0.23).

language modalities – such as image and video captioning (Donahue et al., 2017; Fang et al., 2015; Vinyals et al., 2015; Wang et al., 2018; Yu et al., 2016), visual question answering (VQA) (Antol et al., 2015; Yang et al., 2016), and visual dialog (Das et al., 2017) – VLN additionally requires agents to plan their actions, move, and dynamically respond to changes in their visual field.

Photo-realistic simulations for VLN are especially promising: they retain messy, real world complexity and can draw on pre-trained models and rich data about the world, but do not require investment in and maintenance of physical robots and spaces for them. Given this, we focus on the Room-to-Room (R2R) task (Anderson et al., 2018b). Despite significant recent progress on R2R since its introduction (Fried et al., 2018; Ma et al., 2019; Wang et al., 2019), the structure of the dataset and current evaluation metrics greatly diminish the im-

---

[*]Authors contributed equally.

portance of language understanding for the task. The core problems are that paths in R2R are all direct-to-goal shortest paths and metrics are mostly based on goal completion rather than fidelity to the described path. To address this, we define a new metric, Coverage weighted by Length Score (CLS), and compose path pairs of R2R to create Room-for-Room (R4R), an algorithmically produced extension of R2R. Figure 1 illustrates path composition and the scores of two agent paths for both CLS and Success weighted by Path Length (SPL), a metric recently proposed by Anderson et al. (2018a). In the example, an agent which ignores the language but gets to the goal receives a perfect SPL score.

Language is not irrelevant for R2R. Thomason et al. (2019) ablate visual and language inputs and find that withholding either from an action sampling agent reduces performance on unseen houses. Also, the generated instructions in the augmented paths of Fried et al. (2018) improved performance for several models. However, while many of these augmented instructions have clear starting *or* ending descriptions, the middle portions are often disconnected from the path they are paired with (see Huang et al. (2019) for in depth analysis of augmented path instructions). That these low-fidelity augmented instructions improve results indicates that current metrics are insensitive to instruction fidelity. Our new CLS metric measures how closely an agent's trajectory conforms with the entire reference path, not just goal completion.

Because the reference paths in R2R are all direct-to-goal, the importance of the actual journey taken from start to finish is diminished; as a result, fidelity between instructions and their corresponding paths is harder to evaluate. In longer, twistier paths, the importance of not always going directly to the goal becomes much clearer. We take advantage of the fact that the original R2R data contains many paths that have goals that coincide with the start points of other paths. By concatenating pairs of paths and their corresponding instructions, we create longer paths that allow us to better gauge the ability of an agent to stick to the path as described. With this data, Reinforced Cross-modal Matching models (Wang et al., 2019) that use CLS as a reward signal dramatically improve not only CLS (from 20.4% for the agent with goal-oriented rewards to 34.6%), but navigation error also reduces from 8.45m to 8.08m on the the Validation Unseen dataset. Furthermore, we find that the agent with goal-oriented rewards obtains the same CLS (20.4) on R4R regardless of whether the full instruction or only the last five tokens are provided to it. In contrast, the CLS-rewarded agent drops from CLS of 34.6 to 25.3 when given only the last five tokens.

2 Extending R2R to create R4R

Instructions such as "*Turn left, walk up the stairs. Enter the bathroom.*" are easy for people but challenging for computational agents. Agents must segment instructions, set sub-goals based on understanding them and ground the language and their actions in real world objects and dynamics. An agent may need expectations for how spatial scenes change when turning. Additionally, it must recognize visual and environmental features that indicate it has entered or encountered something referred to as "*the bathroom*" and know to stop.

2.1 Room-to-Room (R2R)

Room-to-Room (R2R) supports visually-grounded natural language navigation in a photo-realistic environment (Anderson et al., 2018b). R2R consists of an environment and language instructions paired to reference paths. The environment defines a graph where nodes are possible positions an agent may inhabit. Edges indicate that a direct path between two nodes is navigable. For each node, R2R provides an egocentric panoramic view. All images are collected from buildings and house interiors. The paths paired with language instructions are composed by sequences of nodes in this graph.

For data collection, starting and goal nodes are sampled from the graph and the shortest path between those nodes is taken, provided it is no shorter than 5m and contains between 4 and 6 edges. Each path has 3 associated natural language instructions, with an average length of 29 words and a total vocabulary of 3.1k words. Apart from the training set, the dataset includes two validation sets and a test set. One of the validation sets includes new instructions on environments overlapping with the training set (Validation Seen), and the other is entirely disjoint from the training set (Validation Unseen). Fried et al. (2018) propose a *follower* model which is trained using student forcing, where actions are sampled from the agent's decisions, but supervised using the action that takes the agent closest to the goal. During inference, the *follower* generates candidate paths which are then scored by a *speaker* model. The *speaker* model was also used
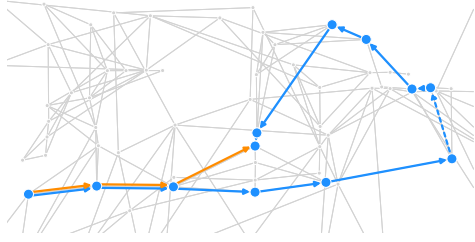
Figure 2: An example of an extended path in the R4R dataset, where the dotted blue arrow connects two blue paths with solid arrows, corresponding to the instructions *"Make a left down at the narrow hall beside the office and walk straight to the exit door. Go out the door and wait."* and *"Turn around and enter the bedroom. Walk to the other side of the room and turn left. Walk into the doorway leading out and stop."*. The shortest-to-goal path from the starting point is shown in orange.

		#samples	PL(R)	$d(r_1, r_{|R|})$
R2R	Train	14039	9.91	9.91
	Val. seen	1021	10.2	10.2
	Val. unseen	2249	9.50	9.50
R4R	Train	233613	20.6	10.5
	Val. seen	1035	20.4	11.1
	Val. unseen	45162	20.2	10.1

Table 1: Comparison of R2R to R4R. PL(R) represents the mean path length of the reference paths and $d(r_1, r_{|R|})$ is mean length of the shortest-to-goal path.

for creating an augmented dataset that is used as an extension of training data by the *follower* model as well as by many subsequently published models. Wang et al. (2019) train their agents using policy gradients. At every step, the agent is rewarded for getting closer to the target location (extrinsic reward) as well as for choosing an action that reduces cycle-reconstruction error between instruction generated by a matching critic and ground-truth instruction (intrinsic reward). In both papers, there is little analysis presented about the generative models.

Recently, Anderson et al. (2018a) pointed out weaknesses in the commonly used metrics for evaluating the effectiveness of agents trained on these tasks. A new metric, Success weighted by Path Length (SPL) was proposed that penalized agents for taking long paths. Any agent using beam search (e.g. Fried et al. (2018)), is penalized heavily by this metric. There have also been concerns about structural biases present in these datasets which may provide *hidden* shortcuts to agents training on these problems. Thomason et al. (2019) presented an analysis on R2R dataset, where the trained agent continued to perform surprisingly well in the absence of language inputs.

2.2 Room-for-Room (R4R)

Due to the process by which the data are generated, all R2R reference paths are shortest-to-goal paths. Because of this property, conformity to the instructions is **decoupled** from reaching the desired destination – and this short-changes the language perspective. In a broader scope of reference paths, the importance of following language in-

structions in their entirety becomes clearer, and proper evaluation of this conformity can be better studied. Additionally, the fact that the largest path in the dataset has only 6 edges exacerbates the challenge of properly evaluating conformity. This motivates the need for a dataset with larger and more diverse reference paths.

To address the lack of path variety, we propose a data augmentation strategy[1] that introduces long, twisty paths without additional human or low-fidelity machine annotations (e.g. those from Fried et al. (2018)). Existing paths in the dataset can be extended by joining them with other paths that start within some threshold of where they end. Formally, two paths $A = (a_1, a_2, \cdots, a_{|A|})$ and $B = (b_1, b_2, \cdots, b_{|B|})$ are joined if $d(a_{|A|}, b_1) < d_{th}$. The resulting extended paths are thus $R = (a_1, \cdots, a_{|A|}, c_1, \cdots, c_{|C|}, b_1, \cdots, b_{|B|})$, where $C = (c_1, c_2, \cdots, c_{|C|})$ is the shortest path between $a_{|A|}$ and b_1. (If $a_{|A|} = b_1$, C is empty.)

Each combination of instructions corresponding to paths A and B is included in R4R. Since each path maps to multiple human-annotated instructions, each extended path will map to $N_A \cdot N_B$ joined instructions, where N_A and N_B are the number of annotations associated with paths A and B, respectively. Figure 2 shows an example of an extended path and the corresponding instructions, compared to the shortest-to-goal path.

3 Evaluation Metrics in VLN

Historically, the performance of VLN models has been evaluated with respect to the objective of reaching the goal location. The nature of the path an agent takes, however, is of clear practical importance: it is undesirable for any robotic agent in the physical world to reach the destination by taking a

---

[1]R2R-to-R4R code is at https://github.com/google-research/google-research/tree/master/r4r
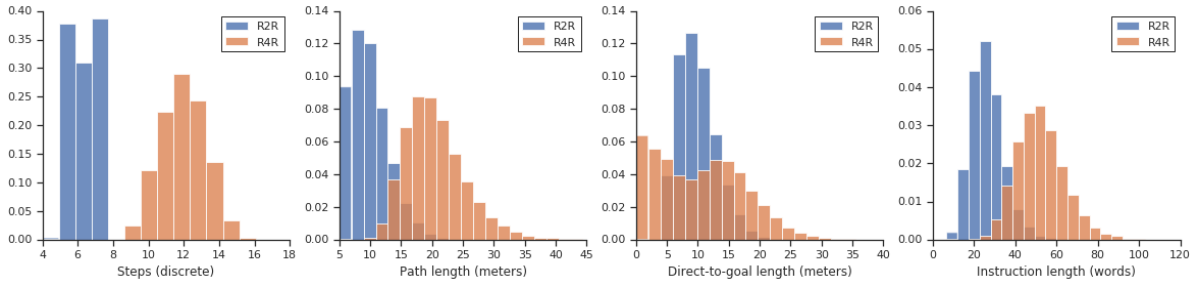
Figure 3: From left to right, the distribution of the number of steps, path lengths, direct-to-goal path lengths and instruction lengths in the original R2R and extended R4R datasets.

different path than what it was instructed to follow; failure to comply with instructions might lead to navigating unwanted and potentially dangerous locations. Here, we propose a series of desiderata for VLN metrics and introduce Coverage weighted by Length Score (CLS). Table 2 provides a high level summary of this section's contents.

3.1 Desiderata

Commonly, navigation tasks are defined in a discrete space: the environment determines a graph where each node is a position the agent could be in and each edge between two nodes represents that there is a navigable step between them. Let the predicted path $P = (p_1, p_2, p_3, ..., p_{|P|})$ be the sequence of nodes visited by the agent and reference path $R = (r_1, r_2, r_3, ..., r_{|R|})$ be the sequence of nodes in the reference trajectory. Generally, $p_1 = r_1$, since in many VLN tasks, the agent begins at the reference path's start node. The following desiderata characterize metrics that gauge the fidelity of P with respect to R rather than just goal completion. Throughout the paper, we refer to the subsequent desired properties as *Desideratum* (i).

(1) *Path similarity measure.* Metrics should characterize a notion of similarity between a predicted path P and a reference path R. This implies that metrics should depend on all nodes in P and all nodes in R, which contrasts with many common metrics which only consider the last node in the reference path (see Section 3.2). Metrics should penalize deviations from the reference path, even if they lead to the same goal. This is not only prudent, as agents might wander around undesired terrain if this is not enforced, but also explicitly gauges the fidelity of the predictions with respect to the provided language instructions.

(2) *Soft penalties.* Metrics should penalize differences from the reference path according to a soft notion of dissimilarity that depends on distances in the graph. This ensures that larger discrepancies are penalized more severely than smaller ones and that metrics should not rely only on dichotomous views of intersection. For instance, a predicted path that has no intersection to the reference path, but follows it closely, as illustrated in Figure 1 should not be penalized too severely.

(3) *Unique optimum.* Metrics should yield a perfect score if and only if the reference and predicted paths are an exact match. This ensures that the perfect score is unambiguous: the reference path R is therefore treated as a golden standard. No other path should have the same or higher score as the reference path itself.

(4) *Scale invariance.* Metrics should be consistent over different datasets.

(5) *Computational tractability.* Metrics should be pragmatic, allowing fast automated evaluation of performance in navigation tasks.

3.2 Existing Navigation Metrics

Table 2 defines previous navigation metrics and how they match our desiderata. We denote by $d(n, m)$ the shortest distance between two nodes along the edges of the graph and $d(n, P) = \min_{p \in P} d(n, p)$ the shortest distance between a node and a path. All distances are computed along the edges of the graph determined by the environment, which are not necessarily equal to the euclidean distance between the nodes.

Path Length (PL) measures the total length of the predicted path, which has the optimal value equal to the length of the reference path. *Navigation Error (NE)* measures the distance between the last node in the predicted path and the last reference path node. *Oracle Navigation Error (ONE)* measures the shortest distance from any node in the predicted path to the last reference path node. *Success Rate (SR)* measures how often the last node in the predicted path is within a threshold distance

Metric	↑↓	Definition	Desiderata coverage				
			(1)	(2)	(3)	(4)	(5)
Path Length (PL)	-	$\sum_{1\le i<|P|} d(p_i, p_{i+1})$		✓			✓
Navigation Error (NE)	↓	$d(p_{|P|}, r_{|R|})$		✓			✓
Oracle Navigation Error (ONE)	↓	$\min_{p\in P} d(p, r_{|R|})$		✓			✓
Success Rate (SR)	↑	$\mathbb{1}[\mathrm{NE}(P,R) \le d_{th}]$				✓	✓
Oracle Success Rate (OSR)	↑	$\mathbb{1}[\mathrm{ONE}(P,R) \le d_{th}]$				✓	✓
Success weighted by PL (SPL)	↑	$\mathrm{SR}(P,R) \cdot \dfrac{d(p_1, r_{|R|})}{\max\{\mathrm{PL}(P), d(p_1, r_{|R|})\}}$		✓		✓	✓
Success weighted by Edit Distance (SED)	↑	$\mathrm{SR}(P,R) \left(1 - \dfrac{\mathrm{ED}(P,R)}{\max\{|P|,|R|\}-1}\right)$	✓		✓	✓	✓
Coverage weighted by LS (CLS)	↑	$\mathrm{PC}(P,R) \cdot \mathrm{LS}(P,R)$	✓	✓	✓	✓	✓

Table 2: Definition and desiderata coverage of navigation metrics.

d_{th} of the last reference path node. *Oracle Success Rate (OSR)* measures how often any node in the predicted path is within a threshold distance d_{th} of the last node in the reference path.

Success weighted by Path Length (SPL) (Anderson et al., 2018a) takes into account both Success Rate and the normalized path length. It was proposed as a single summary measure for navigation tasks. Note that the agent should maximize this metric, and it is only greater than 0 if the success criteria was met. While this metric is ideally suited when the evaluating whether the agent successfully reached the desired destination, it does not take into account any notion of similarity between the predicted and reference trajectories and fails to take into account the intermediary nodes in the reference path. As such, it violates *Desideratum* (1). Since there could exist more than one path with optimal length to the desired destination, it also violates *Desideratum* (3).

Success weighted by Edit Distance (SED) (Chen et al., 2019) is based on the edit distance $\mathrm{ED}(P,R)$ between the two paths, equal to the Levenshtein distance between the two sequences of actions $A_P = ((p_1, p_2), (p_2, p_3), ..., (p_{|P|-1}, p_{|P|}))$ and $A_R = ((r_1, r_2), (r_2, r_3), ..., (r_{|R|-1}, r_{|R|}))$. The Levenshtein distance is the minimum number of edit operations (insertion, deletion and substitution of actions) that can transform path A_R into A_P. Similarly to SPL, SED is also multiplied by $\mathrm{SR}(P,R)$, so only paths that meet the success criteria receive a score greater than 0. This metric naturally satisfies *Desideratum* (1), (3) and (4). Further, it is possible to compute it using dynamic programming in $O(|P||R|)$, further satisfying *Desideratum*
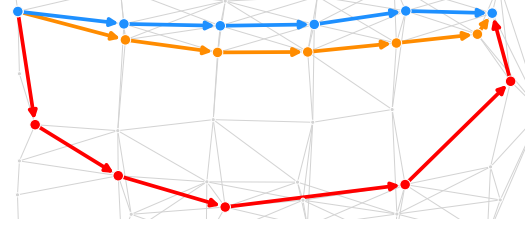

Figure 4: With respect to the blue path, SED yields zero for both the orange and red paths, while CLS yields a score of 0.89 for orange and 0.48 for red.

(5). *Desideratum* (2), however, is left unsatisfied, as SED does not take into account *how* two actions differ from each other (considering, for instance, the graph distance between their end nodes), but only if they are the same or not. This subtle but important difference is illustrated in Figure 4.

3.3 Coverage weighted by Length Score

We introduce *Coverage weighted by Length Score (CLS)* as a single summary measure for VLN. CLS is the product of the Path Coverage (PC) and Length Score (LS) of the agent's path P with respect to reference path R:

$$\mathrm{CLS}(P,R) = \mathrm{PC}(P,R) \cdot \mathrm{LS}(P,R) \quad (1)$$

PC replaces SR as a non-binary measure of how well the reference path is covered by the agent's path. It is the average coverage of each node in the reference path R with respect to path P:

$$\mathrm{PC}(P,R) = \frac{1}{|R|} \sum_{r\in R} \exp\left(-\frac{d(r,P)}{d_{th}}\right) \quad (2)$$

where $d(r, P) = \min_{p \in P} d(r, p)$ is the distance to reference path node r from the nearest node in P. The coverage contribution for each node r is an exponential decay of this distance. ($1/d_{th}$ is a decay constant to account for graph scale.)

LS compares the predicted path length $\mathrm{PL}(P)$ to EPL, the expected optimal length given R's coverage of P. If say, the predicted path covers only half of the reference path (i.e., $PC = 0.5$), then we expect the optimal length of the predicted path to be half of the length of the reference path. As a result, EPL is given by:

$$\mathrm{EPL}(P, R) = \mathrm{PC}(P, R) \cdot \mathrm{PL}(R) \qquad (3)$$

LS for a predicted path P is optimal only if $\mathrm{PL}(P)$ is equal to the expected optimal length – it is penalized when the predicted path length is shorter or longer than the expected path length:

$$\mathrm{LS}(P, R) = \frac{\mathrm{EPL}(P, R)}{\mathrm{EPL}(P, R) + |\mathrm{EPL}(P, R) - \mathrm{PL}(P)|}$$
$$(4)$$

There is a clear parallel between the terms of CLS and SPL. CLS replaces success rate, the first term of SPL, with path coverage, a continuous indicator for measuring how well the predicted path covered the nodes on the reference path. Unlike SR, PC is sensitive to the intermediary nodes in the reference path R. The second term of SPL penalizes the path length $\mathrm{PL}(P)$ of the predicted path against the optimal (shortest) path length $d(p_1, r_{|R|})$; CLS replaces that with length score LS, which penalizes the agent path length $\mathrm{PL}(P)$ against EPL, the expected optimal length for its coverage of R.

CLS naturally covers *Desideratum* (1) and (2). Assuming that the reference path is acyclic and that $p_1 = r_1$, i.e., reference and predicted path start at the same node, *Desideratum* (3) is also satisfied. Additionally, CLS also covers *Desideratum* (4) because PC and LS are both invariant to the graph scale (due to the term d_{th}). Finally, the distances from each pair of nodes in the graph can be pre-computed using Dijkstra's algorithm (Dijkstra, 1959) for each node, resulting in a complexity of $O(EV + V^2 \log(V))$, where V and E are the number of vertices and edges in the graph, respectively. $PC(P, R)$ can be computed in $O(|P||R|)$, and $LS(P, R)$ can be computed in $O(|P| + |R|)$, making CLS satisfy *Desideratum* (5).

4 Agent

We reimplement the Reinforced Cross-Modal Matching (RCM) agent of Wang et al. (2019) and extend it to use a reward function based on both CLS (Section 3.3) as well as success rate.

4.1 Navigator

The *reasoning navigator* of Wang et al. (2019) learns a policy π_θ over parameters θ that map the natural language instruction $\mathcal{X}$ and the initial visual scene v_1 to a sequence of actions $a_{1..T}$. At time step t, the agent state is modeled using a LSTM (Hochreiter and Schmidhuber, 1997) that encodes the trajectory of past visual scenes and agent actions, $h_t = LSTM([v_t; a_{t-1}], h_{t-1})$, where v_t is the output of visual encoder as described below.

Language Encoder Language instructions $\mathcal{X} = x_{1..n}$ are initialized with pre-trained GloVe word embeddings (Pennington et al., 2014) that are fine-tuned during training. We restrict the GloVe vocabulary to tokens that occur at least five times in the instruction data set. All out of vocabulary tokens are mapped to a single OOV identifier. Using a bidirectional recurrent network (Schuster and Paliwal, 1997) we encode the instruction into language contextual representations $w_{1..n}$.

Visual Features As in Fried et al. (2018), at each time step t, the agent perceives a 360-degree panoramic view of its surroundings from the current location. The view is discretized into m view angles ($m = 36$ in our implementation, 3 elevations x 12 headings at 30-degree intervals). The image at view angle i, heading angle ϕ and elevation angle θ is represented by a concatenation of the pre-trained CNN image features with the 4-dimensional orientation feature [sin ϕ; cos ϕ; sin θ; cos θ] to form $v_{t,i}$. The visual encoder pools the representation of all view angles $v_{t,1..m}$ using attention over the previous agent state h_{t-1}.

$$v_t = \mathrm{Attention}(h_{t-1}, v_{t,1..m}) \qquad (5)$$

The actions available to the agent at time t are denoted as $u_{t,1..l}$, where $u_{t,j}$ is the representation of navigable direction j from the current location obtained similarly to $v_{t,i}$ (Fried et al., 2018). The number of available actions, l, varies for different locations, since nodes in the graph have different number of connections.

Action Predictor As in Wang et al. (2019), the

model predicts the probability p_k of each navigable direction k using a bilinear dot product.

$$p_k = \text{softmax}([h_t; c_t^{text}; c_t^{visual}]W_c(u_{t,k}W_u)^T) \quad (6)$$

$$c_t^{text} = \text{Attention}(h_t, w_{1..n}) \quad (7)$$

$$c_t^{visual} = \text{Attention}(c_t^{text}, v_{t,1..m}) \quad (8)$$

4.2 Learning

Training is performed using two separate phases, (1) behavioral cloning (Bain and Sammut, 1999; Wang et al., 2019; Daftry et al., 2016) and (2) REINFORCE policy gradient updates (Williams, 1992).

As is common in cases where expert demonstrations are available, the agent's policy is initialized using behavior cloning to constrain the learning algorithm to first model state-action spaces that are most relevant to the task, effectively warm starting the agent with a good initial policy. No reward shaping is required during this phase as behavior cloning corresponds to solving the following maximum-likelihood problem,

$$\max_\theta \sum_{(s,a)\in\mathcal{D}} \log \pi_\theta(a|s) \quad (9)$$

where $\mathcal{D}$ is the demonstration data set.

After warm starting the model with behavioral cloning, we obtain standard policy gradient updates by sampling action sequences from the agent's behavior policy. As in standard policy gradient updates, the model is optimized by minimizing the loss function $\mathcal{L}^{PG}$ whose gradient is the negative policy gradient estimator (Williams, 1992).

$$\mathcal{L}^{PG} = -\hat{\mathbb{E}}_t[\log \pi_\theta(a_t|s_t)\hat{A}_t] \quad (10)$$

where the expectation $\hat{\mathbb{E}}_t$ is taken over a finite batch of sample trajectories generated by the agent's stochastic policy π_θ. To reduce variance, we scale the gradient using the advantage function $\hat{A}_t = R_t - \hat{b}_t$. ($R_t = \sum_{i=t}^\infty \gamma^{i-t} r_i$ is the observed γ-discounted episodic return and $\hat{b}_t$ is the estimated value of the agent's current state at time t.)

The models are trained using mini-batch gradient descent. Our experiments show that interleaving behavioral cloning and policy gradient training phases improves performance on the validation set. Specifically we interleaved each policy gradient update batch with K behaviour cloning batches, with the value of K decaying exponentially, such that the training strategy asymptotically becomes only policy gradient updates.

4.3 Reward

For consistency with the established benchmark (Wang et al., 2019), we implemented a dense *goal-oriented* reward function that optimizes the *success rate* metric. This includes an immediate reward at time step t in an episode of length T, given by:

$$r(s_t, a_t) = \begin{cases} d(s_t, r_{|R|}) - \\ d(s_{t+1}, r_{|R|}) & \text{if } t < T \\ \mathbb{1}[d(s_T, r_{|R|}) \le d_{th}] & \text{if } t = T \end{cases} \quad (11)$$

where $d(s_t, r_{|R|})$ is the distance between s_t and target location $r_{|R|}$, $\mathbb{1}[\cdot]$ is the indicator function, d_{th} is the maximum distance from $r_{|R|}$ that the agent is allowed to terminate for success.

To incentivize the agent to not only reach the target location but also to conform to the reference path, we also train our agents with following *fidelity-oriented* sparse reward:

$$r(s_t, a_t) = \begin{cases} 0 & \text{if } t < T \\ \mathbb{1}[d(s_T, r_{|R|}) \le d_{th}] + \\ CLS(s_{1...T}, R) & \text{if } t = T \end{cases} \quad (12)$$

where R is the reference path in the dataset associated with the instruction $\mathcal{X}$. This rewards actions that are consistent both with reaching the goal and following the path corresponding to the language instructions. It is worth noting here that, similar to Equation 11, a relative improvement in CLS can be added as a reward-shaping term for time steps $t < T$, however empirically we did not find noticeable difference in the performance of agents trained with or without the shaping term. For simplicity, all of the experiments involving *fidelity-oriented* reward use the sparse reward in Equation 12.

5 Results

We obtain the performance of models trained under two training objectives. The first is *goal oriented* (Equation 11): agents trained using this reward are encouraged to pursue only the last node in the reference path. The second is *fidelity oriented* (Equation 12): agents trained using this reward receive credit not only for reaching the target location successfully but also for conforming to the reference path. We report the performance on standard metrics (PL, NE, SR, SPL) as well as the new CLS metric.

To further explore the role of language, we perform ablation studies, where agents are trained using the full language instructions and evaluated on partial (last 5 tokens) or no instructions. With no instructions, the agent only has the full visual input, similar to the unimodal ablation studies of Thomason et al. (2019). To eliminate the effect observed due to distribution shift during evaluation and preserve the length distribution of the input instructions, we further conducted studies where agents are given arbitrary instructions from the validation set, with the reference path remaining unaltered. We observed that experiments with arbitrary instruction had similar results to studies where instructions where fully removed.

On the R4R dataset, the *fidelity oriented* agent significantly outperforms the *goal oriented* agent ($> 14\%$ absolute improvement in CLS), demonstrating that including CLS in the reward signal successfully produces better conformity to the reference trajectories. Furthermore, on Validation Unseen, when all but the last 5 tokens of instructions are removed, the *goal oriented* agent yields the same CLS as with the full instructions, while the *fidelity oriented* agent suffers significantly, decaying from 34.6% to 25.3%. This indicates that including fidelity measurements as reward signals improve the agent's reliance on language instructions–thereby better keeping the **L** in VLN.

5.1 R2R Performance

Table 3 summarizes the experiments on R2R.[2] There are not major differences between *goal oriented* and *fidelity oriented* agents, highlighting the problematic nature of R2R paths with respect to instruction following: essentially, rewards that only take into account the goal implicitly signals path conformity—by the construction of the dataset itself. As a result, an agent optimized to reach the

target destination may incidentally appear to be conforming to the instructions. The results shown in Section 5.2 further confirm this hypothesis by training and evaluate *goal oriented* and *target oriented* agents on R4R dataset.

As evidenced by the ablation studies, models draw some signal from the language instructions. However, having the last five tokens makes up for a significant portion of the gap between no instructions and full instructions, again highlighting problems with R2R and the importance in R2R of identifying the right place to stop rather than following the path. The performance of both the agents degrade in similar proportions when instructions are partially or fully removed.

Finally, as expected, the SPL metric appears consistent with CLS on R2R, since all reference paths are shortest-to-goal. As highlighted in Section 5.2, this breaks in settings where paths twist and turn.

5.2 R4R Performance

Table 4 shows the results on R4R. Overall, the scores for all model variants on R4R are much lower than R2R, which highlights the additional challenge of following longer instructions for longer paths. Most importantly, the *fidelity oriented* agent significantly outperforms the *goal oriented* agent for both CLS and navigation error, demonstrating the importance of both measuring path fidelity and using it to guide agent learning.

On the experiments, the *goal oriented* agent continues to exploit biases and the underlying structure in the environment to reach the goal. When the instructions are removed during evaluation, the agent's performance on the CLS metric barely degrades, showing that the agent does not rely significantly on the instructions for its performance. In contrast, the *fidelity oriented* agent learns to pursue conformity to the reference path, which in turn requires attending more carefully to the instructions. When instructions are removed during evaluation, performance of the *fidelity oriented* agent degrades considerably on the CLS metric. In fact, the *fidelity oriented* agent performs better on CLS metric without instructions as the *goal oriented* agent performs with the full instructions.

Furthermore, we highlight that historically dominant metrics are ineffective – even misleading – for measuring agents' performance: for instance, especially for reference paths that begin and end at close locations, SPL is a poor measure of suc-

---

[2]Our *goal oriented* results match the RCM benchmark on validation unseen but are lower on validation seen. We suspect this is due to differences in implementation details and hyper-parameter choices.

[3]For the random evaluation, we first sample the number of edges in the trajectory from the distribution of number of edges in the reference paths of the training dataset. Then, for each node, we uniformly sample between its neighbors and move the agent there. We report the average metrics for 1 million random trajectories.

[4]As in Wang et al. (2019), we report the performance of Speaker-Follower model from Fried et al. (2018) that utilizes panoramic action space and augmented data but no beam search (pragmatic inference) for a fair comparison.

[5]We report the performance of the RCM model without intrinsic reward as the benchmark.

		Validation Seen					Validation Unseen				
#	Model	PL	NE↓	SR↑	SPL↑	CLS↑	PL	NE↓	SR↑	SPL↑	CLS↑
0	Random[3]	10.4	9.82	5.0	3.7	29.4	9.32	9.32	5.2	4.0	29.0
1	Speaker-Follower (Fried et al., 2018)[4]	-	3.36	66.4	-	-	-	6.62	35.5	-	-
2	RCM (Wang et al., 2019)[5]	12.1	**3.25**	**67.6**	-	-	15.0	6.01	40.6	-	-
3	Speaker-Follower	15.5	4.98	50.1	40.1	54.8	15.2	6.36	35.3	28.1	42.9
4	RCM, *goal oriented*	13.7	4.48	55.3	47.9	**61.1**	14.8	**6.00**	**41.1**	32.7	47.4
5	last 5 tokens	16.9	7.35	26.5	22.2	39.0	15.1	8.16	22.2	17.2	35.1
6	no instructions	21.1	7.78	22.3	11.6	27.5	17.7	8.69	13.0	9.4	26.1
7	RCM, *fidelity oriented*	12.2	4.63	57.3	**50.7**	60.2	13.2	6.38	40.8	**35.1**	**50.9**
8	last 5 tokens	13.4	8.08	27.8	23.5	42.4	14.4	8.29	23.2	17.7	35.5
9	no instructions	20.1	8.95	18.2	8.8	24.8	20.5	8.76	14.3	6.2	22.7

Table 3: Results on R2R Validation Seen and Validation Unseen sets. Rows 0 and 3-9 shows numbers from our implementations. SR, SPL and CLS are reported as percentages and NE and PL in meters.

		Validation Seen					Validation Unseen				
#	Model	PL	NE↓	SR↑	SPL↑	CLS↑	PL	NE↓	SR↑	SPL↑	CLS↑
0	Random[3]	21.8	11.4	13.1	2.0	23.1	23.6	10.4	13.8	2.2	22.3
1	Speaker-Follower	15.4	5.35	51.9	**37.3**	46.4	19.9	8.47	23.8	**12.2**	29.6
2	RCM, *goal oriented*	24.5	**5.11**	**55.5**	32.3	40.4	32.5	8.45	**28.6**	10.2	20.4
3	last 5 tokens	29.5	8.73	26.4	12.4	35.1	29.5	9.04	23.4	4.5	20.4
4	no instructions	32.3	9.50	20.7	8.0	33.3	34.0	9.45	19.0	2.3	17.4
5	RCM, *fidelity oriented*	18.8	5.37	52.6	30.6	**55.3**	28.5	**8.08**	26.1	7.7	**34.6**
6	last 5 tokens	17.1	8.88	24.8	11.7	39.3	25.5	8.52	18.9	5.6	25.3
7	no instructions	12.7	10.5	12.1	5.4	37.2	22.8	9.41	15.5	4.9	23.0

Table 4: Results on R4R Validation Seen and Validation Unseen sets (see Section 2). SR, SPL and CLS are reported as percentages and NE and PL in meters.

cess since it assumes the optimal path length is the shortest distance between the starting and ending positions (as illustrated in Figure 1, for example). This is particularly noticeable from the results: the *goal oriented* agent gets better SPL scores than the *fidelity oriented* agent, even when it has massively poorer performance on conformity (CLS).

6 Conclusion

The CLS metric, R4R, and our experiments provide a better toolkit for measuring the impact of better language understanding in VLN. Furthermore, our findings suggests ways that future datasets and metrics for judging agents should be constructed and set up for evaluation. The R4R data itself clearly still has considerable headroom: our reimplementation of the RCM model gets only 34.6 CLS on paths in R4R's Validation Unseen houses. Keeping in mind that humans have an average navigation error of 1.61 in R2R (Anderson et al., 2018b), the average navigation error of 8.08 meters for R4R by our best agent leaves plenty of headroom. Future agents will need to make effective use of language and its connection to the environment to both drive CLS up and bring NE down in R4R.

We expect path fidelity to not only be interesting with respect to grounding language, but to be crucial for many VLN-based problems. For example, future extensions of VLN will likely involve games (Baldridge et al., 2018) where the instructions being given take the agent around a trap or help it avoid opponents. Similar constraints could hold in search-and-rescue human-robot teams (Kruijff et al., 2014; Kruijff-Korbayov et al., 2016), where the direct path could take a rolling robot into an area with greater danger of collapse. In such scenarios, going straight to the goal could be literally deadly to the robot or agent.

References

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

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

S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. 2015. VQA: Visual question answering. In *2015 IEEE International Conference on Computer Vision (ICCV)*. pages 2425–2433.

Michael Bain and Claude Sammut. 1999. A framework for behavioural cloning. In *Machine Intelligence 15, Intelligent Agents [St. Catherine's College, Oxford, July 1995]*. Oxford University, Oxford, UK, UK, pages 103–129.

Jason Baldridge, Tania Bedrax-Weiss, Daphne Luong, Srini Narayanan, Bo Pang, Fernando Pereira, Radu Soricut, Michael Tseng, and Yuan Zhang. 2018. Points, paths, and playscapes: Large-scale spatial language understanding tasks set in the real world. In *Proceedings of the First International Workshop on Spatial Language Understanding*. Association for Computational Linguistics, New Orleans, pages 46–52.

Yonatan Bisk, Kevin Shih, Yejin Choi, and Daniel Marcu. 2018. Learning interpretable spatial operations in a rich 3d blocks world. In *Proceedings of the Thirty-Second Conference on Artificial Intelligence (AAAI-18)*. New Orleans, USA.

Valts Blukis, Dipendra Misra, Ross A. Knepper, and Yoav Artzi. 2018. Mapping navigation instructions to continuous control actions with position visitation prediction. In *Proceedings of the Conference on Robot Learning*.

Howard Chen, Alane Suhr, Dipendra Misra, and Yoav Artzi. 2019. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *Conference on Computer Vision and Pattern Recognition*.

Volkan Cirik, Yuan Zhang, and Jason Baldridge. 2018. Following formulaic map instructions in a street simulation environment. *NIPS Visually Grounded Interaction and Language Workshop* .

Shreyansh Daftry, J. Andrew Bagnell, and Martial Hebert. 2016. Learning transferable policies for monocular reactive MAV control. In *International Symposium on Experimental Robotics*. Springer, pages 3–11.

Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M.F. Moura, Devi Parikh, and Dhruv Batra. 2017. Visual Dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Harm de Vries, Kurt Shuster, Dhruv Batra, Devi Parikh, Jason Weston, and Douwe Kiela. 2018. Talk the walk: Navigating new york city through grounded dialogue. *CoRR* abs/1807.03367.

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

J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell. 2017. Long-term recurrent convolutional networks for visual recognition and description. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39(4):677–691.

H. Fang, S. Gupta, F. Iandola, R. K. Srivastava, L. Deng, P. Dollr, J. Gao, X. He, M. Mitchell, J. C. Platt, C. L. Zitnick, and G. Zweig. 2015. From captions to visual concepts and back. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. pages 1473–1482.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-Follower models for Vision-and-Language Navigation. In *Neural Information Processing Systems (NeurIPS)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.* 9(8):1735–1780.

Haoshuo Huang, Vihan Jain, Harsh Mehta, Jason Baldridge, and Eugene Ie. 2019. Multi-modal discriminative model for vision-and-language navigation. In *Proceedings of the Combined Workshop on Spatial Language Understanding and Grounded Communication for Robotics (SpLU-RoboNLP-2019)*. Association for Computational Linguistics, Minneapolis.

G.J.M. Kruijff, Ivana Kruijff-Korbayova, Shanker Keshavdas, Benoit Larochelle, Miroslav Janicek, Francis Colas, Ming Liu, Franois Pomerleau, Roland Siegwart, Mark Neerincx, Rosemarijn Looije, Nanja Smets, Tina Mioch, Jurriaan Diggelen, Fiora Pirri, Mario Gianni, Federico Ferri, Matteo Menna, Rainer Worst, and Vaclav Hlavac. 2014. Designing, developing, and deploying systems to support human-robot teams in disaster response. *Advanced Robotics* 28.

I. Kruijff-Korbayov, L. Freda, M. Gianni, V. Ntouskos, V. Hlav, V. Kubelka, E. Zimmermann, H. Surmann, K. Dulic, W. Rottner, and E. Gissi. 2016. Deployment of ground and aerial robots in earthquake-struck amatrice in italy (brief report). In *2016 IEEE*

International Symposium on Safety, Security, and Rescue Robotics (SSRR). pages 278–279.

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

Matt Macmahon, Brian Stankiewicz, and Benjamin Kuipers. 2006. Walk the talk: Connecting language, knowledge, action in route instructions. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*. pages 1475–1482.

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

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pages 1532–1543.

M. Schuster and K.K. Paliwal. 1997. Bidirectional recurrent neural networks. *Trans. Sig. Proc.* 45(11):2673–2681.

Pararth Shah, Marek Fiser, Aleksandra Faust, Chase Kew, and Dilek Hakkani-Tur. 2018. FollowNet: Robot navigation by following natural language directions with deep reinforcement learning. In *Third Machine Learning in Planning and Control of Robot Motion Workshop at ICRA*.

D. Skočaj, A. Vrečko, M. Mahnič, M. Janíček, G.-J. M. Kruijff, M. Hanheide, N. Hawes, J. L. Wyatt, T. Keller, K. Zhou, M. Zillich, and M. Kristan. 2016. An integrated system for interactive continuous learning of categorical knowledge. *Journal of Experimental & Theoretical Artificial Intelligence* 28:823–848.

Jesse Thomason, Daniel Gordon, and Yonatan Bisk. 2019. Shifting the baseline: Single modality performance on visual navigation & QA. In *Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Jesse Thomason, Jivko Sinapov, Raymond Mooney, and Peter Stone. 2018. Guiding exploratory behaviors for multi-modal grounding of linguistic descriptions. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* pages 3156–3164.

Xin Wang, Wenhu Chen, Jiawei Wu, Yuan-Fang Wang, and William Yang Wang. 2018. Video captioning via hierarchical reinforcement learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* pages 4213–4222.

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

Edward C. Williams, Nakul Gopalan, Mina Rhee, and Stefanie Tellex. 2018. Learning to parse natural language to grounded reward functions with weak supervision. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*. pages 1–7.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning* 8(3):229–256.

Claudia Yan, Dipendra Misra, Andrew Bennett, Aaron Walsman, Yonatan Bisk, and Yoav Artzi. 2018. CHALET: Cornell House Agent Learning Environment. *CoRR* abs/1801.07357.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alexander J. Smola. 2016. Stacked attention networks for image question answering. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* pages 21–29.

Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. 2016. Video paragraph captioning using hierarchical recurrent neural networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. pages 4584–4593.