Transferable Representation Learning in Vision-and-Language Navigation

Haoshuo Huang*   Vihan Jain*   Harsh Mehta   Alexander Ku   Gabriel Magalhaes

Jason Baldridge   Eugene Ie

Google Research

1600 Amphitheatre Parkway, Mountain View, CA 94043, United States

{haoshuo, vihan, harshm, alexku, gamaga, jridge, eugeneie}@google.com

Abstract

Vision-and-Language Navigation (VLN) tasks such as Room-to-Room (R2R) require machine agents to interpret natural language instructions and learn to act in visually realistic environments to achieve navigation goals. The overall task requires competence in several perception problems: successful agents combine spatio-temporal, vision and language understanding to produce appropriate action sequences. Our approach adapts pre-trained vision and language representations to relevant in-domain tasks making them more effective for VLN. Specifically, the representations are adapted to solve both a cross-modal sequence alignment and sequence coherence task. In the sequence alignment task, the model determines whether an instruction corresponds to a sequence of visual frames. In the sequence coherence task, the model determines whether the perceptual sequences are predictive sequentially in the instruction-conditioned latent space. By transferring the domain-adapted representations, we improve competitive agents in R2R as measured by the success rate weighted by path length (SPL) metric.

1. Introduction

Vision-and-Language Navigation (VLN) requires computational agents to represent and integrate both modalities and take appropriate actions based on their content, their alignment and the agent's position in the environment. VLN datasets have graduated from simple virtual environments [26] to photo-realistic environments, both indoors [2] and outdoors [10, 7, 19]. To succeed, VLN agents must internalize the (possibly noisy) natural language instruction, plan action sequences, and move in environments that dynamically change what is presented in their visual fields. These challenging settings bring simulation-based VLN work closer to real-world, language-based interaction with robots [28].

Along with these challenges come opportunities: for ex-
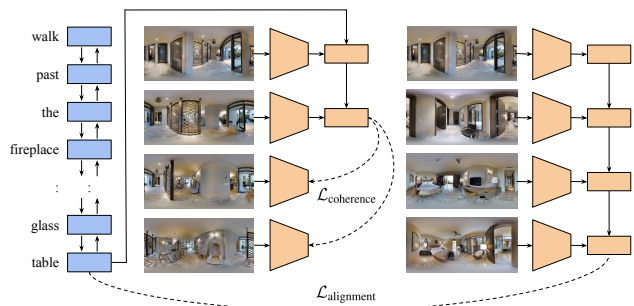
---

* Authors contributed equally.

Figure 1: To overcome the scarcity of high-quality human-annotated data, we propose auxiliary tasks, CMA and NVS, that can be created by simple and effective negative mining. The representations learned by a model trained on both the tasks simultaneously, with a combined loss $\alpha\mathcal{L}_{\text{alignment}}+(1-\alpha)\mathcal{L}_{\text{coherence}}$, are transferred over to agents learning the VLN navigation task. The RCM agent [39] so trained outperforms the existing published state-of-the-art agents.

ample, pre-trained linguistic and visual representations can be injected into agents before training them on example instructions-path pairs. Work on the Room-to-Room (R2R) dataset [2] typically uses GloVe word embeddings [30] and features from deep image networks like ResNet [17] trained on ImageNet [31]. Associations between the input modalities are based on co-attention, with text and visual representations conditioned on each other. Since a trajectory spans multiple time steps, the visual context is often modeled using recurrent techniques like LSTMs [20] that combine features from the current visual field with historical visual signals and agent actions. The fusion of both modalities constitutes the agent's belief state. The agent relies on this belief state to decide which action to take, often relying on reinforcement learning techniques like policy gradient [41].

Unfortunately, due to domain shift, the pre-trained models are poor matches for R2R's instructions and visual observa-

tions. Furthermore, human-annotated data is expensive to collect and there are relatively few instruction-path pairs (e.g. R2R has just 7,189 paths with instructions). This greatly reduces the expected benefit of fine-tuning [16, 45] on the navigation task itself. Our contribution is to define auxiliary, discriminative learning tasks that exploit the environment *before* agent training. Our high-quality augmentation strategy adapts the out-of-domain pre-trained representations and allows the agent to focus on learning how to act rather than struggling to bridge representations *while* learning how to act. It furthermore allows us to rank and better exploit the outputs of generative strategies used previously [14].

We present three main contributions. First, we define two in-domain auxiliary tasks: Cross-Modal Alignment (CMA), which involves assessing the fit between a given instruction-path pair, and Next Visual Scene (NVS), which involves predicting latent representations of future visual inputs in the path. Neither task requires additional human annotated data as they are both trained with cheap negative mining techniques following Huang *et al.* [22]. Secondly, we propose methods to train models on the two tasks: *alignment-based* similarity scores for CMA and contrastive predictive coding [36] for NVS. A model trained on CMA and NVS is not only able to learn cross-modal alignments, but is also able to correctly differentiate between high-quality and low-quality instruction-path pairs in the augmented data introduced by Fried *et al.* [14]. Finally, we show that representations learned by this model can be transferred to two competitive navigation agents, Speaker-Follower [14] and Reinforced Cross-Modal [39], to outperform their previously established results. We also found that our domain-adapted agent outperforms the known state-of-the-art agent at the time by 5% absolute measure in SPL.

2. Related Work

Vision-and-Language Grounding There is much prior work in the intersection of computer vision and natural language processing [42, 23, 27, 21]. A highly related class of tasks centers around generating captions for images and videos [12, 13, 37, 38, 44]. In Visual Question Answering [3, 43] and Visual Dialog [9], models generate single-turn and multi-turn responses by co-grounding vision and language. In contrast to these tasks, VLN agents are embodied in the environment and must combine language, scene, and spatio-temporal understanding.
Embodied Agent Navigation Navigation in realistic 3D environments has also received increased interest recently [35, 18, 29, 46]. Advances in vision-and-language navigation have accelerated with the introduction of the Room-to-Room (R2R) dataset and associated attention-based sequence-to-sequence baseline [2]. Fried *et al.* [14] used generative approaches to augment the instruction-path pairs and proposed a modified beam search for VLN. Wang *et al.*

[39] introduced innovations around multi-reward RL with imitation learning and co-grounding in the visual and text modality. While the two approaches reused pre-trained vision and language modules directly in the navigation agent, our contribution shows that these pre-trained components can be further enhanced by adapting them to related auxiliary tasks before employing them in a VLN agent.

3. The Room-to-Room Dataset

The Room-to-Room (R2R) dataset [2] is based on 90 houses from the Matterport3D environments [6] each defined by an undirected graph. The nodes are locations where egocentric photo-realistic panoramic images are captured and the edges define the connections between locations. The dataset consists of language instructions paired with reference paths, where each path is a sequence of graph nodes. Each path is associated with 3 natural language instructions collected using Amazon Mechanical Turk with an average token length of 29 from a dictionary of 3.1k unique words. Paths collected are longer than 5m and contain 4 to 6 edges. The dataset is split into a training set, two validation sets and a test set. One validation set includes new instructions on environments overlapping with the training set (Validation Seen), and the other is entirely disjoint from the training set (Validation Unseen). Evaluation on the validation unseen set and the test set assess the agent's full generalization ability. Metrics for assessing agents performance include:

- *Path Length (PL)* measures the total length of the predicted path. (The reference path's length is optimal.)

- *Navigation Error (NE)* measures the distance between the last nodes in the predicted and the reference paths.

- *Success Rate (SR)* measures how often the last node in the predicted path is within some threshold distance d_{th} of the last node in the reference path.

- *Success weighted by Path Length (SPL)* [1] measures whether the SR success criteria was met, weighted by the normalized path length.

SPL is the best metric for ranking agents as it takes into account the path taken, not just whether goal was reached [1]. This is evident with (invalid) entries on the R2R leaderboard that use beam search often achieving high SR but low SPL because they wander all around before stopping.

4. Mining Negative Paths

VLN tasks are composed of instruction-path pairs, where a path is a sequence of connected locations along with their corresponding perceptual contexts. The core task is to train agents to follow the provided instructions. However, auxiliary tasks could help adapt out-of-domain language and

vision representations to be relevant to the navigation domain. We follow two principles in designing these auxiliary tasks: they should not involve any additional human annotations and they should use and update representations needed for downstream navigation tasks.

The crux of our auxiliary tasks is the observation that the given human generated instructions are specific to the paths described. Given the diversity and relative uniqueness of the properties of different rooms and the trajectories of different paths, it is highly unlikely that the original instruction will correspond well to automatically mined negative paths. As such, given a visual path and a high quality human generated instruction, it is easy to create various incorrect paths by random path sampling or random walks from start or end nodes, to name a few. For a given instruction-path pair, we sample negatives by keeping the same instruction but altering the path sequence in one of three ways.

- *Path Substitution (PS)*: randomly pick other paths from the same environment as negatives.

- *Random Walks (RW)*: sample random paths of the same length as the original path that either (1) start at the same location and end sufficiently far from the original path or (2) end at the same location and start sufficiently far from the original path. We use a threshold of 5 meters to make sure the path has significant difference.

- *Partial Reordering (PR)*: keep the first and last nodes in the path fixed and randomly shuffle the rest.

These three strategies create increasingly more challenging negative examples. PS pairs have only incidental connection between the text and the perceptual sequence, RW pairs share one or the other end point, and PR pairs have the same perceptual elements in a new (and incoherent) order.

5. Representation Learning

Using the mined negative paths, we train models for two auxiliary tasks that exploit the data in complementary ways. The first is a two-tower model [15, 33] with a cross-modal alignment module. This model produces similarity scores that reflect the semantic similarity between visual and language sequences. The second is a model that optimizes pairwise sequence coherence by predicting latent representations of future visual scenes, conditioned on the language sequence and a partial visual sequence. We furthermore train these models on both tasks with a combined loss. This fine tunes the representations to domain-specific language and interior environments relevant to the R2R dataset, and associates language to the visual scenes the agent will experience during the full navigation problem.

5.1. Task 1: Cross-Modal Alignment (CMA)

An agent's ability to navigate a visual environment using language instructions is closely associated with its capacity to align semantically similar concepts across the two modalities. Given an instruction like "*Turn right and move forward around the bed, enter the bathroom and wait there.*", the agent should match the word *bed* with a location on the path that has a bed in the agent's egocentric view; doing so will help orient the agent and allow it to better follow further instructions. To this end, we create a cross-modal alignment task (denoted as CMA) that involves discriminating positive instruction-path pairs from negative pairs. The discriminative model is based on an *alignment-based* similarity score that encourages the model to map perceptual and textual signals in two sequences.

5.2. Task 2: Next Visual Scene (NVS)

Research in sensory and motor processing suggests that the human brain predicts (anticipates) future states in order to assist decision making [11, 5]. Similarly, agents can benefit if they learn to predict expected future states given the current context at a given point in the course of navigation. While it is challenging to predict high-dimensional future states, methods like Contrastive Predictive Coding (CPC) [36] circumvent this by working in lower dimensional latent spaces. With CPC, we add a probabilistic contrastive loss to our adaptation model. This induces a latent space that captures visual information useful for predicting future visual observations, enabling the visual network to adapt to the R2R environment. In the NVS task, the model's current state is used to predict the latent space representation of future k steps (in this work, we use $k = 1, 2$). The negatives from CMA are used as negatives to compute the InfoNCE [36] loss during training (see next section for details).

5.3. Model Architecture

For consistency with the navigation agent model (Sec. 6), we use a two-tower architecture to encode the two sequences, with one tower encoding the token sequence in the instruction and the other tower encoding the visual sequence.

Language Encoder. Instructions $\mathcal{X} = x_1, x_2, ..., x_n$ are initialized with pre-trained GloVe word embeddings [30]. These embeddings are fine-tuned to solve the auxiliary tasks and transferred to the agent to be further fine-tuned to solve the VLN challenge. We restrict the GloVe vocabulary to tokens that occur at least five times in the training instructions. All out-of-vocabulary tokens are mapped to a single out-of-vocabulary identifier. The token sequence is encoded using a bi-directional LSTM [32] to create H^X following:

$$H^X = [h_1^X; h_2^X; ...; h_n^X] \tag{1}$$

$$h_t^X = \sigma(\overrightarrow{h}_t^X, \overleftarrow{h}_t^X) \tag{2}$$

$$\overrightarrow{h}_t^X = LSTM(x_t, \overrightarrow{h}_{t-1}^X) \tag{3}$$

$$\overleftarrow{h}_t^X = LSTM(x_t, \overleftarrow{h}_{t+1}^X) \tag{4}$$

where the σ function is used to combine the output of forward and backward LSTM layers.

Visual Encoder. As in Fried *et al.* [14], at each time step t, the agent perceives a 360-degree panoramic view at its current location. The view is discretized into k view angles ($k = 36$ in our implementation, 3 elevations by 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 [$\sin \phi$; $\cos \phi$; $\sin \theta$; $\cos \theta$] to form $v_{t,i}$. The visual input sequence $\mathcal{V} = v_1, v_2, ..., v_m$ is encoded using a LSTM to create H^V following:

$$H^V = [h_1^V; h_2^V; ...; h_m^V] \tag{5}$$

$$h_t^V = LSTM(v_t, h_{t-1}^V) \tag{6}$$

where $v_t = \text{Attention}(h_{t-1}^V, v_{t,1..k})$ is the attention-pooled representation of all view angles using previous agent state h_{t-1} as the query.

Training Loss. For CMA, the alignment-based similarity score is computed as follows:

$$A = H^X(H^V)^T \tag{7}$$

$$\{c\}_{l=1}^{l=X} = \text{softmax}(A^l) \cdot A^l \tag{8}$$

$$\text{score} = \text{softmin}(\{c\}_{l=1}^{l=X}) \cdot \{c\}_{l=1}^{l=X} \tag{9}$$

where $(.)^T$ is matrix transpose transformation, A is the alignment matrix whose dimensions are $[n, m]$ and A^l is the l-th row vector in A. Eq. 8 corresponds to taking a softmax along the columns and summing the columns. This amounts to column-wise content-based pooling. Then we apply the softmin operation along the rows and sum the rows up to obtain a scalar in Eq. 9. Intuitively, maximizing this score for positive instruction-path pairs encourages the learning algorithm to construct the best worst-case sequence alignment between the two sequences in the latent space. The training objective for CMA is to minimize the cross entropy loss $\mathcal{L}_{\text{alignment}}$.

The InfoNCE [36] loss for NVS is computed as follows:

$$\mathcal{L}_{\text{coherence}} = -\mathbb{E}_F \left[\log \frac{f(v_{t+k}, h_t^V)}{\sum_{v_j \in F} f(v_j, h_t^V)} \right] \tag{10}$$

$$f(v_{t+k}, h_t^V) = \exp(v_{t+k}^T W_k h_t^V) \tag{11}$$

where v_{t+k} is the latent representation of visual input at time step $t + k$, h_t^V is the visual-encoder LSTM's output at time step t which summarizes all $v_{\leq t}$, W_k are learnable parameters which are different for different values of k (we choose $k = 1, 2$ in our experiments). For a given h_t^V, there is exactly one positive sample in the set F, the negative samples can be chosen from negative instruction-path pairs as mined in Sec. 4. The loss in Eq. 10 is the categorical cross-entropy of classifying the positive sample correctly.

Finally, the model is trained to minimize the combined loss $\alpha \mathcal{L}_{\text{alignment}} + (1 - \alpha)\mathcal{L}_{\text{coherence}}$.

6. Navigation Agent

For comparisons with established models, we reimplemented the Speaker Follower agent of Fried *et al.* [14] (denoted as SF agent from hereon) and Reinforced Cross-Modal Matching agent of Wang *et al.* [39] (denoted as RCM agent from hereon) for our experiments.

6.1. Navigator

The navigator 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 $a_{1..T}$. The language and visual encoder of the navigator are the same as described in Sec. 5.3. The actions available to the agent at time t are denoted as $u_{t,1..l}$, where $u_{t,j}$ is the representation of the navigable direction j from the current location obtained similarly to $v_{t,i}$ [14]. The number of available actions, l, varies per location, since graph node connectivity varies. As in [39], the model predicts the probability p_d of each navigable direction d using a bilinear dot product:

$$p_d = \text{softmax}([h_t^V; c_t^{\text{text}}; c_t^{\text{visual}}]W_c(u_{t,d}W_u)^T) \tag{12}$$

$$c_t^{\text{text}} = \text{Attention}(h_t^V, h_{1..n}^X) \tag{13}$$

$$c_t^{\text{visual}} = \text{Attention}(c_t^{\text{text}}, v_{t,1..k}) \tag{14}$$

6.2. Learning

The SF agent is trained using *student forcing* [14] where actions are sampled from the model during training, and supervised using a shortest-path action to reach the goal.

For the RCM agent, learning is performed in two separate phases, (1) behavioral cloning [4, 39, 8] and (2) REINFORCE policy gradient updates [41]. The agent's policy is initialized using behavior cloning to maximally use the available expert demonstrations. This phase constrains 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:

$$\max_\theta \sum_{(s,a)\in\mathcal{D}} \log \pi_\theta(a|s) \qquad (15)$$

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

Once the model is initialized to a reasonable policy with behavioral cloning, we further update the model via standard policy gradient updates by sampling action sequences from the agent's behavior policy. As in standard policy gradient updates, the model minimizes the loss function $\mathcal{L}^{\text{PG}}$ whose gradient is the negative policy gradient estimator [41]:

$$\mathcal{L}^{\text{PG}} = -\hat{\mathbb{E}}_t[\log \pi_\theta(a_t|s_t)\hat{A}_t] \qquad (16)$$

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. Furthermore, for variance reduction, we scale the gradient using the advantage function $\hat{A}_t = R_t - \hat{b}_t$ where $R_t = \sum_{i=t}^\infty \gamma^{i-t} r_i$ is the observed γ-discounted episodic return and $\hat{b}_t$ is the estimated value of agent's current state at time t. Similar to [39], the immediate reward at time step t in an episode of length T is given by:

$$r(s_t, a_t) = \begin{cases} d(s_t, r_{|R|}) - d(s_{t+1}, r_{|R|}) & \text{if } t < T \\ \mathbb{1}[d(s_T, r_{|R|}) \le d_{th}] & \text{if } t = T \end{cases} \qquad (17)$$

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 it to be considered successful.

The models are trained using mini-batch gradient descent. For RCM agent, 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.

7. Results

7.1. Experimental Setup

In our experiments, we use a 2-layer bi-directional LSTM for the instruction encoder where the size of LSTM cells is 256 units in each direction. The inputs to the encoder are 300-dimensional embeddings initialized using GLoVe and fine-tuned during training. For the visual encoder, we use a 2-layer LSTM with a cell size of 512 units. The encoder inputs are image features derived as mentioned in Sec. 5.3. The cross-modal attention layer size is 128 units. To train the model on auxiliary tasks, we use Momentum optimizer

PS	PR	RW	AUC
✓			64.5
	✓		60.5
		✓	63.1
✓	✓		72.1
	✓	✓	66.0
✓		✓	70.8
✓	✓	✓	72.0

Table 1: Results on training in different combinations of datasets and evaluating against validation dataset containing PR and RW negatives only.

with a learning rate of 10^{-2} that decays at a rate of 0.8 every 0.5 million steps. The SF navigation agent is trained using Momentum optimizer while RCM agent is trained using Adam optimizer with learning rate decaying at a rate of 0.5 every 0.2 million steps. We use a learning rate of 10^{-5} during agent training if the agent is warm-started with pre-trained components of the model trained on auxiliary tasks, otherwise we use learning rate of 10^{-4}.

7.2. Training on Auxiliary Tasks

Recently, Fried *et al.* [14] introduced an augmented dataset (referred to as `Fried-Augmented` from now on) that is generated by using a speaker model and they show that the models trained with both the original data and the machine-generated augmented data improves agent success rates. On manual inspection, we found that while many paths in `Fried-Augmented` have clear starting *or* ending descriptions, the middle segments of the instructions are often noisy and have little connection to the path they are meant to describe. Here we show that our model trained on CMA is able to differentiate between high-quality and low-quality instruction-path pairs in `Fried-Augmented`.

In line with the original R2R dataset [2], we create three splits for each of the negative sampling strategies defined in Section 5 – a training set from paths in R2R *train* split, a validation seen set from paths in R2R *validation seen* and a validation unseen set from paths in R2R *validation unseen* split. The paths in the original R2R dataset are used as positives and there are 10 negatives for each positive with 4 of those negatives sampled using PS and 3 each using RW and PR respectively. A model trained on the task CMA learns to differentiate aligned instruction-path pairs from the misaligned pairs. We also studied the three negative sampling strategies summarized in Table 1.

Scoring generated instructions. We use this trained model to rank all the paths in `Fried-Augmented` and train the RCM agent on different portions of the data. Table 2 gives the performance when using the best 1% versus the worst 1%, and likewise for the best and worst 2%. Using high-quality examples–as judged by the model–outperforms

Dataset size	Strategy	Validation Seen				Validation Unseen			
		PL	NE ↓	SR ↑	SPL ↑	PL	NE ↓	SR ↑	SPL ↑
1%	Top	11.1	8.5	21.2	17.6	11.2	8.5	20.4	16.6
	Bottom	10.7	9.0	16.3	13.1	10.8	8.9	15.4	14.1
2%	Top	11.7	7.9	25.5	21.0	11.3	8.2	22.3	18.5
	Bottom	14.5	9.1	17.7	12.7	11.4	8.4	17.5	14.1

Table 2: Results for Validation Seen and Validation Unseen, when trained with a small fraction of `Fried-Augmented` ordered by scores given by model trained on CMA. SPL and SR are reported as percentages and NE and PL in meters.

Figure 2: Alignment matrix (Eq. 7) for model trained on the dataset containing (a) PS, PR, RW negatives (b) PS negatives only. Note that darker means higher alignment.

the ones trained using low-quality examples. Note that the performance is low in both cases because none of the original human-created instructions were used—what is important is the relative performance between examples judged higher or lower. This clearly indicates that the model scores instruction-path pairs effectively.

Visualizing Cross-Modal Alignment. Fig. 2 gives the alignment matrix A (Eq. 7) from the model trained on CMA for a given instruction-path pair to try to better understand how well the model learns to align the two modalities as hypothesized. As a comparison point, we also plot the alignment matrix for a model trained on the dataset with PS negatives only. While scoring PR and RW negatives may require carefully aligning the full sequence in the pair, it is easier to score PS negatives by just attending to first or last locations on the path. It is expected that the model trained on the dataset containing only PS negatives will exploit these easy-to-find patterns in negatives and make predictions without carefully attending to full instruction-path sequence.

The figure shows the difference between cross-modal alignment for the two models. While there is no clear alignment between the two sequences for the model trained with PS negatives only (except maybe towards the end of sequences, as expected), there is a visible diagonal pattern in the alignment for the model trained on all negatives in CMA. In fact, there is appreciable alignment at the correct positions in the two sequences, *e.g.*, the phrase *exit the door* aligns with the image(s) in the path containing the object *door*, and similarly for the phrase *enter the bedroom*.

Improvements from Adding Coherence Loss. Finally we show that training a model on CMA and NVS simultaneously improves the model's performance when evaluated on CMA alone. The model is trained using combined loss $\alpha\mathcal{L}_{\text{alignment}} + (1-\alpha)\mathcal{L}_{\text{coherence}}$ with $\alpha = 0.5$ and is evaluated on its ability to differentiate incorrect instruction-path pairs from correct ones. As noted earlier, PS negatives are easier to discriminate, therefore, to keep the task challenging, the validation sets were limited to contain validation splits from

Training	Val. Seen	Val. Unseen
CMA	82.6	72.0
NVS	63.0	62.1
CMA + NVS	**84.0**	**79.2**

Table 3: AUC performance when the model is trained on different combinations of the two tasks and evaluated on the dataset containing only PR and RW negatives.

PR and RW negative sampling strategies only. The area-under ROC curve (AUC) is used as the evaluation metric. The results in Table 3 demonstrate that adding $\mathcal{L}_{\text{coherence}}$ as auxiliary loss improves the model's performance on CMA by 7% absolute measure.

7.3. Transfer Learning to Navigation Agent

The language and visual encoders in the RCM navigation agent (Sec. 6) are warm-started from the model trained on CMA and NVS simultaneously. The agent is then allowed to train on R2R *train* and Fried-Augmented as other existing baseline models do. We call this agent *ALTR* – to mean an **A**gent initialized by **L**earned **T**ransferable **R**epresentations from auxiliary tasks. The standard testing scenario of the VLN task is to train the agent in seen environments and then test it in previously unseen environments in a zero-shot fashion. There is no prior exploration on the test set. This setting is able to clearly measure the generalizability of the navigation policy, and we evaluate our ALTR agent only under this standard testing scenario.

7.4. Comparison with SOTA

Table 4 shows the comparison of the performance of our ALTR agent to the previous state-of-the-art (SOTA) methods on the test set of the R2R dataset, which is held out as the VLN Challenge. Our ALTR agent significantly outperforms the SOTA at the time on SPL–the primary metric for R2R–improving it by 5% absolute measure, and it has the lowest navigation error (NE). It furthermore ties the other two best models for SR. Compared to RCM, our ALTR agent is able to learn a more efficient policy resulting in shorter trajectories to reach the goal state, as indicated by its lower path length. Figure 3 compares some sample paths from the RCM baseline and our ALTR agent, illustrating that the ALTR agent often stays closer to the true path and does less doubling back compared to the RCM agent.

It is worth noting that the R2R leaderboard has models that use beam-search and/or explore the test environment before submission. For a fair comparison, we only compare against models that, like ours, return exactly one trajectory per sample without pre-exploring the test environment (in accordance with VLN challenge submission guidelines).

We show in the next section that our transfer learning

Model	PL	NE ↓	SR ↑	SPL ↑
Random [2]	9.89	9.79	13.2	12.0
Seq-to-Seq [2]	8.13	7.85	20.4	18.0
Look Before You Leap [40]	9.15	7.53	25.3	23.0
Speaker-Follower [14]	14.8	6.62	35.0	28.0
Self-Monitoring [24]	18.0	5.67	**48.0**	35.0
Reinforced Cross-Modal [39]	12.0	6.12	43.1	38.0
The Regretful Agent [25]	13.7	5.69	**48.0**	40.0
ALTR (Ours)	10.3	**5.49**	48.0	**45.0**

Table 4: Comparison on R2R Leaderboard Test Set. Our navigation model benefits from transfer learned representations and outperforms the known SOTA on SPL. SPL and SR are reported as percentages and NE and PL in meters.
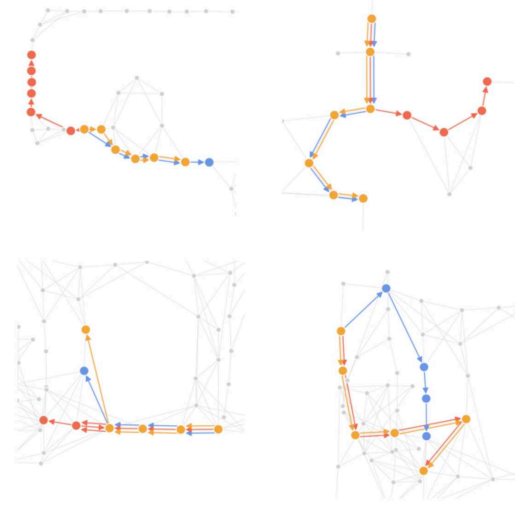

Figure 3: Sample visualizations comparing reference paths (blue), paths from RCM baseline agent (red) and our ALTR agent (orange).

approach improves the Speaker-Follower agent [14]. In general, this strategy is complementary to the improvements from the other agents, so it is likely it would help others too.

7.5. Ablation Studies

The first ablation study analyzes the effectiveness of each task individually in learning representations that can benefit the navigation agent. Since the agent is rewarded for reaching the goal (Eq. 17), we expect SR results to align well with our training objective. Table 5 shows that agents benefit the most when initialized with representations learned on both the tasks simultaneously. When pre-training CMA and NVS jointly, we see a consistent 11-12% improvement in SR for both the SF and RCM agents as well as improvement in agent's path length, thereby also improving SPL. When pre-training CMA only, we see a consistent 8-9% improvement

Method	CMA	NVS	Validation Seen				Validation Unseen			
			PL	NE↓	SR↑	SPL↑	PL	NE↓	SR↑	SPL↑
Speaker-Follower [14]	-	-	-	3.36	66.4	-	-	6.62	35.5	-
RCM[39]	-	-	12.1	3.25	67.6	-	15.0	6.01	40.6	-
Speaker-Follower (Ours)	✗	✗	15.9	4.90	51.9	43.0	15.6	6.40	36.0	29.0
	✓	✗	14.9	5.04	50.2	39.2	16.8	5.85	39.1	26.8
	✗	✓	16.5	5.12	48.7	34.9	18.0	6.30	34.9	20.9
	✓	✓	11.3	4.06	60.8	55.9	14.6	6.06	40.0	31.2
RCM (Ours)	✗	✗	13.7	4.48	55.3	47.9	14.8	6.00	41.1	32.7
	✓	✗	10.2	5.10	51.8	49.0	9.5	5.81	44.8	42.0
	✗	✓	19.5	6.53	34.6	20.8	18.8	6.79	33.7	20.6
	✓	✓	13.2	4.68	55.8	52.7	9.8	5.61	46.1	43.0

Table 5: Ablations on R2R Validation Seen and Validation Unseen sets, showing results in VLN for different combinations of pre-training tasks. SPL and SR are reported as percentages and NE and PL in meters.

Image encoder	Language encoder	Validation Seen				Validation Unseen			
		PL	NE↓	SR↑	SPL↑	PL	NE↓	SR↑	SPL↑
✗	✗	13.7	4.48	55.3	47.9	14.8	6.00	41.1	32.7
✓	✗	15.9	5.05	50.6	38.2	14.9	5.94	42.5	33.1
✗	✓	13.8	4.68	56.3	46.6	13.5	5.66	43.9	35.8
✓	✓	13.2	4.68	55.8	52.7	9.8	5.61	46.1	43.0

Table 6: Ablations showing the effect of adapting (or not) the learned representations in each branch of our RCM agent on Validation Seen and Validation Unseen. SPL and SR are reported as percentages and NE and PL in meters.

in SR for both the SF and RCM agents. When pre-training NVS only, we see a drop in performance. Since there are no cross-modal components to train the language encoder in NVS, training on NVS alone fails to provide a good initialization point for the downstream navigation task that requires cross-modal associations. However, pre-training with NVS and CMA jointly affords the model additional opportunities to improve visual-only pre-training (due to NVS), without compromising cross-modal alignment (due to CMA).

The second ablation analyzes the effect of transferring representations to either of the language and visual encoders. Table 6 shows the results for the RCM agent. The learned representations help the agent to generalize on previously unseen environments. When either of the encoders is warm-started, the agent outperforms the baseline success rates and SPL on validation unseen dataset. In the absence of learned representations, the agent overfits on seen environments and as a result the performance improves on the validation seen dataset. Among the agents that have at least one of the encoders warm-started, the agent with both encoders warm-started has significantly higher SPL (7%+) on the validation unseen dataset.

The results of both the studies demonstrate that the two tasks, CMA and NVS, learn complementary representations which benefit the navigation agent. Furthermore, the agent benefits the most when both the encoders are warm-started from the learned representations.

8. Conclusion

We demonstrate the model trained on two complementary auxiliary tasks, Cross-Modal Alignment (CMA) and Next Visual Scene (NVS), learns visual and textual representations that can be transferred to navigation agents. The transferred representations improve both the SF and RCM agents in key navigation metrics. Our ALTR agent–RCM initialized with domain adapted representations–outperforms published models at the time by 5% absolute measure. We expect our approach to be complementary to the latest state-of-the-art from Tan *et al.* [34].

Similar to our work, there can be other auxiliary tasks that could be designed without requiring any additional human annotations. The scoring model trained on the tasks also has additional capabilities like cross-modal alignment. We expect this could help improve methods that generate additional paired instruction-path pairs. It could also allow us to automatically segment long instruction-path sequences and thus create a curriculum of easy to hard tasks for agent training. For the future, it would be desirable to jointly train the agent with the auxiliary tasks.

References

[1] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir R. Zamir. On evaluation of embodied navigation agents. 2018. `arXiv:1807.06757 [cs.AI]`. 2

[2] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 5, 7

[3] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, and D. Parikh. VQA: Visual Question Answering. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2425–2433, Dec 2015. 2

[4] Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15, Intelligent Agents [St. Catherine's College, Oxford, July 1995]*, pages 103–129, Oxford, UK, UK, 1999. Oxford University. 4

[5] Andreja Bubi, D Cramon, and Ricarda Schubotz. Prediction, cognition and the brain. *Frontiers in human neuroscience*, 4:25, 03 2010. 3

[6] 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. *International Conference on 3D Vision (3DV)*, 2017. 2

[7] Volkan Cirik, Yuan Zhang, and Jason Baldridge. Following formulaic map instructions in a street simulation environment. In *2018 NeurIPS Workshop on Visually Grounded Interaction and Language*, 2018. 1

[8] Shreyansh Daftry, J. Andrew Bagnell, and Martial Hebert. Learning transferable policies for monocular reactive MAV control. *CoRR*, abs/1608.00627, 2016. 4

[9] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M.F. Moura, Devi Parikh, and Dhruv Batra. Visual Dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2

[10] Harm de Vries, Kurt Shuster, Dhruv Batra, Devi Parikh, Jason Weston, and Douwe Kiela. Talk the Walk: Navigating New York City through Grounded Dialogue. *CoRR*, abs/1807.03367, 2018. 1

[11] Massimiliano Di Luca and Darren Rhodes. Optimal perceived timing: Integrating sensory information with dynamically updated expectations. *Scientific reports*, 6:28563, July 2016. 3

[12] J. Donahue, L. A. Hendricks, M. Rohrbach, S. Venugopalan, S. Guadarrama, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):677–691, April 2017. 2

[13] 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. From captions to visual concepts and back. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1473–1482, June 2015. 2

[14] 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 *Neural Information Processing Systems (NeurIPS)*, 2018. 2, 4, 5, 7, 8

[15] Daniel Gillick, Alessandro Presta, and Gaurav Singh Tomar. End-to-end retrieval in continuous space. 2018. `arXiv:1811.08008 [cs.IR]`. 3

[16] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 580–587, 2014. 2

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778, 2016. 1

[18] Sachithra Hemachandra, Felix Duvallet, Thomas M. Howard, Nicholas Roy, Anthony Stentz, and Matthew R. Walter. Learning models for following natural language directions in unknown environments. In *IEEE International Conference on Robotics and Automation, ICRA 2015, Seattle, WA, USA, 26-30 May, 2015*, pages 5608–5615, 2015. 2

[19] Karl Moritz Hermann, Mateusz Malinowski, Piotr Mirowski, Andras Banki-Horvath, and Raia Hadsell Keith Anderson. Learning to follow directions in street view. *CoRR*, abs/1903.00401, 2019. 1

[20] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997. 1

[21] Ronghang Hu, Marcus Rohrbach, and Trevor Darrell. Segmentation from natural language expressions. In *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, pages 108–124, 2016. 2

[22] Haoshuo Huang, Vihan Jain, Harsh Mehta, Jason Baldridge, and Eugene Ie. Multi-modal discriminative model for vision-and-language navigation. In *Proceedings of the Combined Workshop on Spatial Language Understanding (SpLU) and Grounded Communication for Robotics (RoboNLP)*, pages 40–49, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. 2

[23] Andrej Karpathy and Fei-Fei Li. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, pages 3128–3137. IEEE Computer Society, 2015. 2

[24] 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 *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019. 7

[25] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided navigation through progress estimation. 2019. 7

[26] Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. Walk the talk: Connecting language, knowledge, action in route instructions. In *In Proc. of the Nat. Conf. on Artificial Intelligence (AAAI*, pages 1475–1482, 2006. 1

[27] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L. Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *CVPR*, pages 11–20. IEEE Computer Society, 2016. 2

[28] Cynthia Matuszek. Grounded language learning: Where robotics and NLP meet. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 5687–5691. International Joint Conferences on Artificial Intelligence Organization, 7 2018. 1

[29] Piotr Mirowski, Matt Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, and Raia Hadsell. Learning to navigate in cities without a map. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 2419–2430. Curran Associates, Inc., 2018. 2

[30] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics, 2014. 1, 3

[31] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 1

[32] Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Trans. Signal Processing*, 45:2673–2681, 1997. 3

[33] Iulian Vlad Serban, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau. A survey of available corpora for building data-driven dialogue systems: The journal version. *D&D*, 9(1):1–49, 2018. 3

[34] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 2610–2621, 2019. 8

[35] Alexander Toshev, Arsalan Mousavian, James Davidson, Jana Kosecka, and Marek Fiser. Visual representations for semantic target driven navigation. 2018. 2

[36] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018. 2, 3, 4

[37] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2015. 2

[38] Xin Wang, Wenhu Chen, Jiawei Wu, Yuan-Fang Wang, and William Yang Wang. Video captioning via hierarchical re-inforcement learning. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4213–4222, 2018. 2

[39] Xin Wang, Qiuyuan Huang, Asli Çelikyilmaz, 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. *CoRR*, abs/1811.10092, 2018. 1, 2, 4, 5, 7, 8

[40] 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 Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision – ECCV 2018*, pages 38–55, Cham, 2018. Springer International Publishing. 7

[41] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256, May 1992. 1, 4, 5

[42] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 2048–2057, 2015. 2

[43] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alexander J. Smola. Stacked attention networks for image question answering. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21–29, 2016. 2

[44] Haonan Yu, Jiang Wang, Zhiheng Huang, Yi Yang, and Wei Xu. Video paragraph captioning using hierarchical recurrent neural networks. pages 4584–4593, 06 2016. 2

[45] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, pages 818–833, 2014. 2

[46] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J. Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017*, pages 3357–3364, 2017. 2