History Aware Multimodal Transformer for Vision-and-Language Navigation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Vision-and-language navigation (VLN) aims to build autonomous visual agents that follow instructions and navigate in real scenes. To remember previously visited locations and actions taken, most approaches to VLN implement memory using recurrent states. Instead, we introduce a History Aware Multimodal Transformer (HAMT) to incorporate a long-horizon history into multimodal decision making. HAMT efficiently encodes all the past panoramic observations via a hierarchical vision transformer (ViT), which first encodes individual images with ViT, then models the relation between images in a panoramic observation and finally takes into account the temporal relation between panoramas in the history. It, then, jointly combines text, history and current observation to predict the next action. We first train HAMT end-to-end using several proxy tasks including single step action prediction and spatial relation prediction, and then use reinforcement learning to further improve the navigation policy. We evaluate our model on three challenging VLN tasks, namely *fine-grained VLN* (R2R), *long-horizon VLN* (R4R and R2R-Back) and *VLN with high-level instructions* (R2R-Last). HAMT outperforms the state of the art and achieves excellent results in all tasks. We also demonstrate HAMT to be particularly effective for navigation tasks with longer trajectories.

1 Introduction

Vision-and-language navigation (VLN) has recently received growing attention [1, 2, 3, 4, 5]. VLN requires the agents to understand natural language instructions, perceive the visual world, and perform navigation actions to arrive at a target location. A number of datasets have been proposed to support various VLN tasks such as indoor and outdoor navigation with fine-grained instructions [2, 6, 7], language-driven remote object finding [8] and navigation in dialogs [9].

VLN agents are faced with several challenges. First, as opposed to static vision-text grounding [10], the agent continuously receives new visual observations and should align them with instructions. Current works mainly adopt recurrent architectures [5, 6, 11] to encode historical observations and actions within a fixed-size state feature to predict the next action. Such condensed states might be sub-optimal for capturing essential information in extended trajectories [12]. For instance, *"bring the spoon to me"* requires the agent to remember its start location after navigating to the *"spoon"*. Few endeavors [13, 14] construct map-like structures of visited locations as additional memory. Nevertheless, these approaches have strong domain-specific inductive biases and still rely on recurrent states to memorize useful information. As the history plays an important role in environment understanding and instruction grounding, we propose to explicitly encode the history as a sequence of previous actions and observations instead of using recurrent states.

Another VLN challenge concerns the generalizations of agents to new environments that have not been observed during training [4]. One direction is to learn more generic text-image representations.

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

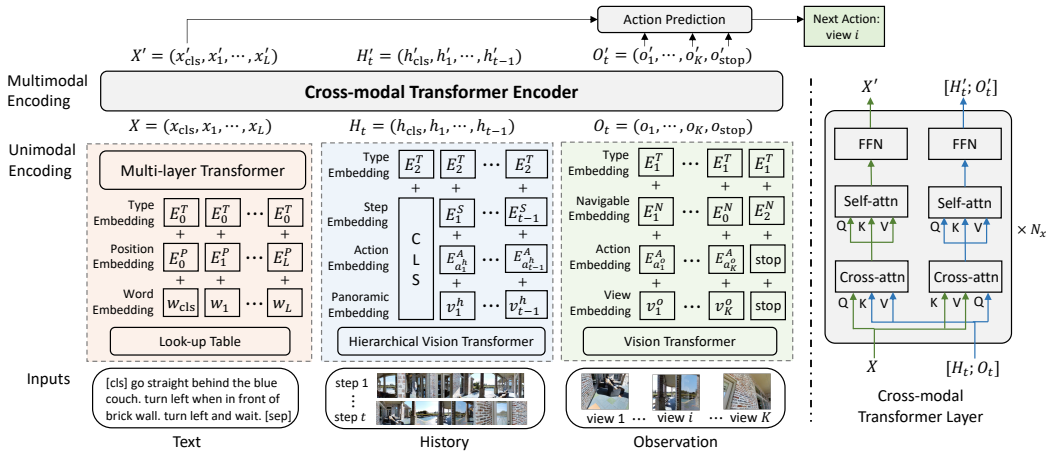


Figure 1: The architecture of History Aware Multimodal Tranformer (HAMT). HAMT jointly encodes textual instruction, full history of previous observations and actions, and current observation to predict the next action.

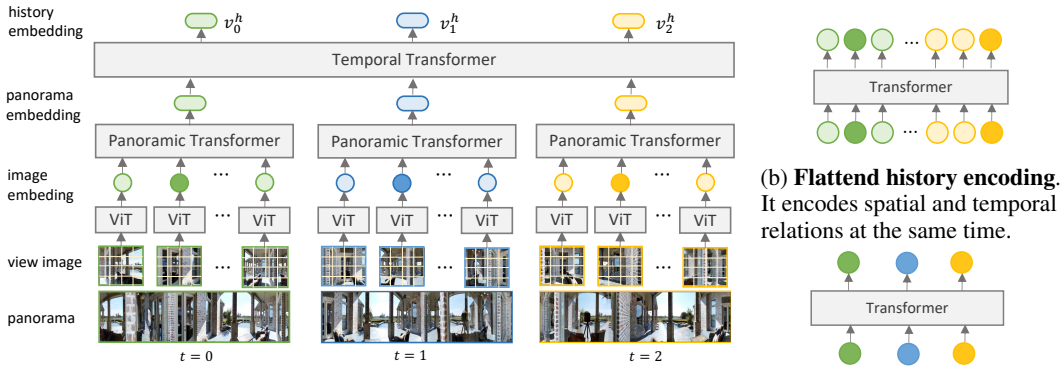
The PRESS model [15] improves language representations with a pretrained BERT encoder [16], and PREVALENT [17] uses pairs of instruction and single-step observations to pretrain a multimodal transformer. Though achieving promising results, these works do not optimize visual representations for the target navigation task. Moreover, the lack of history [17] makes it hard to learn cross-modal alignment and increases the risk of overfitting to training environments. Another direction towards better generalization is to overcome exposure biases [18] due to the discrepancy between training and inference. Different methods have been adopted for VLN including DAGGER [6, 19] and scheduled sampling [15, 20]. Reinforcement Learning (RL) [21, 22] is one of the most effective approach among them, but it is considered unstable to directly train RL with large-scale transformers [23].

To address the above challenges, we propose the History Aware Multimodal Transformer (HAMT), a fully transformer based architecture for multimodal decision making in VLN tasks. As illustrated in Figure 1, HAMT consists of unimodal transformers for text, history and observation encoding, and a cross-modal transformer to capture long-range dependencies of the history sequence, current observation and instruction. Since our history contains a sequence of all previous observations, its encoding is computationally expensive. To resolve complexity issues, we propose a hierarchical vision transformer as shown in Figure 2, which progressively learns representations for a single view, spatial relationships among views within a panorama and, finally, the temporal dynamics across panoramas of the history. To learn better visual representations, we also propose auxiliary proxy tasks for end-to-end HAMT training. Such tasks include single-step action prediction based on imitation learning, self-supervised spatial relationship reasoning, masked language and image predictions and instruction-trajectory matching. We empirically show that our training facilitates the subsequent fine-tuning of our model with RL [24]. We carry out extensive experiments on three distinct VLN tasks, namely *fine-grained VLN* (R2R [6]), *long-horizon VLN* (R4R [3] and our proposed R2R-Back which requires the agent to return back after visiting the target location), and *VLN with high-level instructions* (our proposed R2R-Last). The HAMT model outperforms the state of the art both on seen and unseen environments in all the tasks.

We summarize our contributions as follows: (1) We introduce HAMT, the first fully transformer-based model for VLN relying on a multimodal transformer combining long-horizon history with vision transformers; (2) We train HAMT with auxiliary proxy tasks in an end-to-end fashion and use RL to improve the navigation policy; (3) We validate our method and outperform state of the art in three challenging VLN tasks, while demonstrating larger gains for long-horizon navigation. We will make our codes and new experimental setups publicly available.

2 Related work

Vision-and-Language navigation. Training instruction-following navigation agents has attracted increasing research attention [1, 2, 6, 7, 8, 25]. Anderson *et al.* [6] propose a sequence-to-sequence LSTM baseline for the VLN task. Fried *et al.* [11] extend it with panoramic action space and



(a) **Hierarchical history encoding**. It first encodes individual view images with ViT, then models the spatial relation between images in each panorama, and finally captures the temporal relation between panoramas in the history.

(b) **Flattend history encoding**. It encodes spatial and temporal relations at the same time.

(c) **Temporal-only history encoding**. It only considers temporal relation of oriented views.

Figure 2: A comparison of history encoding methods. Circle nodes in different colors denote view images of panorama at different steps. Darker circle nodes are the oriented view of the agent.

synthesized instructions. To improve cross-modal alignment, the self-monitoring agent [26] proposes co-grounding and progress estimation, and RelGraph [27] uses graphs to model relationships across scene, objects and directions. Reinforcement learning (RL) is typically used to improve navigation policy. The EnvDrop model [22] mixes imitation learning and A3C [24]. The RCM [28] utilizes intrinsic reward of cross-modal matching in REINFORCE algorithm. Wang *et al*. [29] propose to learn rewards via soft expert distillation. Due to the success of transformer [30], recent works explore transformer architectures in VLN. PRESS [15] replaces LSTM instruction encoder with pretrained BERT [16]. SIA [31] uses transformer for single-step multimodal fusion and LSTM for sequential action prediction. PTA [32] is a transformer VLN model using CNNs to extract visual features [33]. Here we propose a full transformer architecture for VLN and train it end-to-end.

Memory-based policy for navigation. LSTMs [34] have been the dominant approach to encode memories for navigation [6, 11, 22, 28]. Condensing all history into one feature vector, however, is prone to the loss of information. Alternative approaches include topological map memory structures [35, 36]. Deng *et al*. [13] use graphs to capture environment layout and enable long-term planing. A similar graph is adopted in [14] with frontier-exploration based decision making. Pursuing more generic memory structure, Fang *et al*. [12] store histories in a sequence encoded with transformer. Recurrent VLN-BERT [5] injects a recurrent unit to encode histories in transformer for VLN. The most similar work to ours is the Episodic Transformer (E.T.) [37]. Differently from [37], we propose a hierarchical encoding of the panoramic observation history and optimize the transformer with end-to-end training.

Multimodal pretraining with transformers. Recent work shows significant progress in vision and language tasks using multimodal pretraining. In particular, transformer architectures such as one-stream [38, 39] and dual-stream [40, 41] achieve state of the art for a number of downstream tasks including visual question answering, image-text retrieval and image captioning. While most previous methods rely on CNN image representations, ViLT [42] follows the work on Vision Transformers (ViT) [43] and trains an and-to-end vision-language transformer using words and image patches as input tokens. A few endeavors [17, 44] explore multimodal pretraining for VLN. PREVALENT [17] pretrains a transformer using instructions and single-step observations without referring to trajectory history. VLN-BERT [44] compares instructions with images along the navigation path but does not support action prediction. Our work presents the first end-to-end trainable VLN transformer that jointly encodes text, history and observations, and sequentially predicts actions.

3 Method

Problem definition The VLN problem [6] is formulated as a partially observable Markov decision process, where future observations are independent of the past conditioning on current state s_t. Given an instruction $\mathcal{W}$ containing a sequence of L words $(w_1, w_2, \cdots, w_L)$, the agent should follow the instruction to move in a connectivity graph to reach the goal location. At each step t, the agent

receives an observation $\mathcal{O}_t$, a panoramic visual input of its surrounding environment. The $\mathcal{O}_t$ consists of K single view images $\mathcal{O}_t \triangleq ([v_1^o; a_1^o], \cdots, [v_K^o; a_K^o])$, where v_i^o is the visual feature of the i-th view and a_i^o denotes the relative angle to face the view (subscript t is omitted for simplicity). There are n navigable views among all the K views, denoted as $\mathcal{O}_t^c \triangleq ([v_1^c; a_1^c], \cdots, [v_n^c; a_n^c])$. We follow the setup in [11] and use $\mathcal{O}_t^c$ as the decision space, so the agent only needs to select a candidate view in $\mathcal{O}_t^c$ at each step. All observations $\mathcal{O}_i$ and performed actions a_i^h before step t form the history $\mathcal{H}_t \triangleq ([\mathcal{O}_1; a_1^h], \cdots, [\mathcal{O}_{t-1}; a_{t-1}^h])$, where a_i^h denotes the turned angles at step i. The goal is to learn a policy π parametrized by Θ to predict the next action based on the instruction, history and the current observation, which is $\pi(a_t|\mathcal{W}, \mathcal{H}_t, \mathcal{O}_t, \mathcal{O}_t^c; \Theta)$.

Unlike dominant recurrent approaches to condense $\mathcal{H}_t$ into a fixed-sized vector, in this section, we present the History Aware Multimodal Transformer (HAMT) that jointly encodes text, long-horizon history, and observation for sequential action prediction. The model architecture is described in Section 3.1. We propose end-to-end VLN training in Section 3.2 to learn unimodal and multimodal representations, and then use RL to fine-tune the navigation policy in Section 3.3.

3.1 HAMT: History Aware Multimodal Transformer

Figure 1 illustrates the model architecture of HAMT. The inputs text $\mathcal{W}$, history $\mathcal{H}_t$ and observation $\mathcal{O}_t$ are first encoded via the corresponding unimodal encoder respectively, and then fed into the cross-modal transformer encoder to capture multimodal relationships.

Text Encoding. For each token i in the instruction $\mathcal{W}$, we embed it as the summation of its word embedding w_i, position embedding E_i^P and type embedding of text E_0^T. Then we employ a transformer with N_L layers to obtain contextual representation x_i following the standard BERT [16].

Observation Encoding. For each view $[v_i^o; a_i^o]$ in the panoramic observation $\mathcal{O}_t$, we first represent the relative angle a_i^o as $E_{a_i^o}^A = (\sin\theta_i, \cos\theta_i, \sin\phi_i, \cos\phi_i)$ where θ_i and ϕ_i are the relative heading and elevation angle. Then the observation embedding o_i is as follows:

$$o_i = \mathrm{LN}(W_v^o v_i^o) + \mathrm{LN}(W_a^o E_{a_i^o}^A) + E_{o_i}^N + E_1^T \tag{1}$$

where W_v^o, W_a^o are learnable weights. The $E_{o_i}^N$ denotes the navigable embedding to differentiate types of views, with E_0^N for non-navigable view, E_1^N for navigable view and E_2^N for stop view (we append a stop token in observation to support stop action). The E_1^T is the type embedding of observation. We omit bias terms for simplicity. The LN denotes layer normalization [45]. Because a_i^o has much lower feature dimensions than v_i^o, we apply LN to balance the encoded a_i^o and v_i^o.

Hierarchical History Encoding. As $\mathcal{H}_t$ consists of all the past panoramic observations $\mathcal{O}_i$ and performed actions a_i^h before step t, it is important to encode $\mathcal{H}_t$ efficiently as context. Figures 2b-2c depict the flattened and temporal-only history encoding approaches used in VLN-BERT [44] and E.T. [37] respectively. The flattened approach treats each view image in $\mathcal{O}_i$ as a token, so the history sequence contains tK tokens. Though it enables to learn relationships among all image views, the computation cost quadratically increases with the sequence length, making it inefficient for long-horizon tasks. In the temporal-only approach, only the oriented view of the agent in each $\mathcal{O}_i$ is taken as inputs instead of the whole panorama, so only t temporal tokens are encoded. However, this approach can lose critical information in past observations. For example, in the instruction *"with the windows on your left, walk through the large room past the sitting areas"*, the object *"window"* does not appear in the oriented view of the agent. Therefore, the encoded history is insufficient to tell whether the agent passed the window or not, making the model confused to take the next action.

In order to balance computational efficiency and information integrity, we propose a hierarchical history encoding approach as illustrated in Figure 2a. It hierarchically encodes image views within each panoramic observation and then the temporal relationships across observations, similar to the factorized spatial-temporal video transformer [46]. For each $\mathcal{O}_i$, its constituent image views are first embeded via Eq (1), and encoded via a transformer with N_h layers to learn spatial relationships with other image views in $\mathcal{O}_i$. We average pool the encoded image views as the panorama embedding, and add it with the oriented image view in residual connection. In this way, $\mathcal{O}_i$ is represented as v_i^h without information loss. Each temporal token in the history sequence is thus computed as:

$$h_i = \mathrm{LN}(W_v^h v_i^h) + \mathrm{LN}(W_a^h E_{a_i^h}^A) + E_i^S + E_2^T \tag{2}$$

Table 1: Comparison of HAMT and previous VLN transformers.

Models	Inputs			Proxy Tasks				
	Text	History	Observation	MLM	MRM	ITM	SAP/SAR	SPREL
PREVALENT [17]	✓		✓	✓			✓	
VLN-BERT [44]	✓	✓		✓	✓	✓		
HAMT (Ours)	✓	✓	✓	✓	✓	✓	✓	✓

where E_i^S denotes the i-th step embedding, E_2^T is the type embedding of history. The computational cost is $O(tK^2+t^2)$, which significantly reduces from $O(t^2K^2)$ in the flattened approach. To be noted, we add a special token `[cls]` to the start of the history sequence to obtain a global representation.

Cross-modal Encoding. We concatenate history and observation as the vision modality, and use cross-modal transformer with N_x layers to fuse features from text, history and observation as shown in the right of Figure 1. The reason of using such dual-stream architecture rather than one-stream is that the length of different modalities can be highly imbalanced, and the dual-stream architecture can balance the importance of intra- and inter-modal relationships by model design [47]. In each cross-modal layer, a vision-text cross-attention is firstly performed for vision modality to attend relevant text information and vice versa for text modality. Then each modality uses self-attention to learn intra-modal relationship such as interaction between observation and history, followed by a fully-connected neural network. Finally, the HAMT model outputs embeddings $X^{'} = (x'_{\text{cls}}, x'_1, \cdots, x'_L), H_t^{'} = (h'_{\text{cls}}, h'_1, \cdots, h'_{t-1}), O_t^{'} = (o'_1, \cdots, o'_K, o'_{\text{stop}})$ for tokens in text, history and observation respectively.

3.2 End-to-end training with proxy tasks

As it is difficult to train large-scale transformers with RL due to sparse supervision [23], we propose to first end-to-end train HAMT via several proxy tasks to learn unimodal and multimodal representation.

Table 1 compares our HAMT with previous VLN transformers PREVALENT [17] and VLN-BERT [44] in inputs and proxy tasks. As neither PREVALENT nor VLN-BERT jointly encodes text, history and observation, limited proxy tasks can be applied in training. Our model instead can take advantage of various proxy tasks to learn cross-modal alignment, spatial and temporal reasoning, and action prediction. Given the input pair $(\mathcal{W}, \mathcal{H}_T)$ where T is the length of full trajectory, we can apply common proxy tasks as in vision-and-language pretraining [40, 44], including Masked Language Modeling (MLM), Masked Region Modeling (MRM) and Instruction Trajectory Matching (ITM). More details are presented in the supplementary material. In the following, we introduce new proxy tasks given the triplet input $(\mathcal{W}, \mathcal{H}_t, \mathcal{O}_t)$ specifically for VLN tasks.

Single-step Action Prediction/Regression (SAP/SAR). The task deploys imitation learning to predict the next action based on instruction, history from expert demonstration and the current observation. We formulate it as a classification and a regression task respectively. In the SAP classification task, we predict action probability for each navigable view in $\mathcal{O}_t^c$ which is $p_t(o'_i) = \frac{\exp(f_{\text{SAP}}(o'_i \odot x'_{\text{cls}}))}{\sum_j \exp(f_{\text{SAP}}(o'_j \odot x'_{\text{cls}}))}$, where f_{SAP} is a two-layer fully-connected network, $\odot$ is element-wise multiplication and x'_{cls} is output embedding of special text token `[cls]`. The objective is to minimize negative log probability of the target view action o'_*: $L_{\text{SAP}} = -\log p_t(o'_*)$. In SAR regression task, we directly predict the action heading and elevation angles based on the text token `[cls]` which is $\hat{\theta}_t, \hat{\phi}_t = f_{\text{SAR}}(x'_{\text{cls}})$. The loss function is $L_{\text{SAR}} = (\hat{\theta}_t - \theta_t)^2 + (\hat{\phi}_t - \phi_t)^2$. The two proxy tasks enable the model to learn how to make action decision conditioning on instruction and contextual history.

Spatial Relationship Prediction (SPREL). Expressions of egocentric spatial relations are frequent in navigational instructions, such as *"walk into the room on your left"*. In order to learn spatial relationship aware representations, we propose the SPREL self-supervised task to predict the relative spatial position of two views in a panorama based on only their visual feature, angle feature or both. Assume $[v_i^o; a_i^o]$ and $[v_j^o; a_j^o]$ are two views in $\mathcal{O}_t$, we randomly zero out v_*^o or a_*^o with probability of 0.3. Their encoded representations after the transformer are o'_i and o'_j, and their relative heading and elevation angles are θ_{ij}, ϕ_{ij}. We then predict $\hat{\theta}_{ij}, \hat{\phi}_{ij} = f_{\text{SPREL}}([o'_i; o'_j])$ where $[;]$ denotes vector concatenation and optimize $L_{\text{SPREL}} = (\hat{\theta}_{ij} - \theta_{ij})^2 + (\hat{\phi}_{ij} - \phi_{ij})^2$. The task helps for egocentric spatial relationship reasoning of the visual observation.

Training Strategy. Instead of directly training the whole HAMT model at once, we propose to progressively train HAMT in two stages. In the first stage, we freeze the vision transformer pretrained on ImageNet [48] and train the rest of the modules which are randomly initialized. This aims to avoid catastrophic forgetting of the pretrained weights in vision transformer. Then we unfreeze the vision transformer and train the whole model end-to-end. The learning rate for the vision transformer is set to be higher than for others modules to avoid vanishing gradients and to speedup convergence.

3.3 Fine-tuning for sequential action prediction

Structure Variants. We present two variants of HAMT for action prediction in the following. 1) MLP action head: we directly reuse the action prediction network f_{SAP} in the SAP task to predict navigable views. We use it as default for VLN tasks. 2) MLP action head based on encoder-decoder structure: the original HAMT model applies cross-modal attention for both vision-to-text and text-to-vision, which is computationally expensive when the instructions are long. Therefore, we remove the cross-modal attention from text to vision. In this way, we separate the cross-modal transformer into an encoder which only takes instruction as input, and a decoder that inputs history and observation as query and attends over encoded text tokens. Please see supplementary material for details.

RL+IM Objective. We combine reinforcement learning (RL) and imitation learning (IM) to fine-tune HAMT for sequential action prediction. The IM relies on the SAP loss defined in Section 3.2 and follows the expert action at each step while RL samples actions according to the policy π. Specifically, we use the Asynchronous Advantage Actor-Critic (A3C) RL algorithm [24]. At each step t, the agent samples an action based on policy π: $\hat{a}_t^h \sim \pi(a_t|\mathcal{W}, \mathcal{H}_t, \mathcal{O}_t, \mathcal{O}_t^c)$ and receives an immediate reward r_t. For non-stop actions, we set r_t as the reduced distance of taking the action to the target and the increased alignment score [3] compared to expert demonstration as defined in [5]; for the stop action, $r_t = 2$ if the agent successfully arrives at the target otherwise -2. A critic network is trained to estimate the value of each state s_t, which is $R_t = \sum_{k=0}^{T-t} \gamma^k r_{t+k}$ where γ is discount factor. We implement it as $V_t = f_{\text{critic}}(x'_{\text{cls}} \odot h'_{\text{cls}})$. As the reward signal favors shortest distance, we empirically find it benefits to combine A3C RL with the imitation learning (IM) weighted by λ, which is:

$$\Theta \leftarrow \Theta + \underbrace{\mu \frac{1}{T} \sum_{t=1}^{T} \nabla_\Theta \log \pi(\hat{a}_t^h; \Theta)(R_t - V_t)}_{\text{Reinforcement Learning (RL)}} + \underbrace{\lambda \mu \frac{1}{T^*} \sum_{t=1}^{T^*} \nabla_\Theta \log \pi(a_t^*; \Theta)}_{\text{Imitation Learning (IM)}} \tag{3}$$

where μ is the learning rate, a_t^* is the expert action at step t of the expert trajectory of length T^*.

4 Experiments

4.1 Experimental setup

Datasets. We evaluate our method on three VLN tasks: *fine-grained VLN* (R2R benchmark [6]); *long-horizon VLN* (R4R [3] and R2R-Back); and *VLN with high-level instructions* (R2R-Last).

- **R2R** [1] builds upon Matterport3D [49] and includes 90 photo-realistic houses with 10,567 panoramic views. It contains 7,189 shortest-path trajectories, each associated with 3 instructions. The dataset is split into train, validation seen, validation unseen and test unseen sets with 61, 56, 11 and 18 houses respectively. The houses in seen split are the same as training split, while houses in unseen splits are different from training.
- **R4R** [3] extends R2R dataset by concatenating two adjacent tail-to-head trajectories in R2R. Therefore, it has longer instructions and trajectories. The trajectories are also less biased as they are not necessarily the shortest-path from start to end location.
- **R2R-Back** is a new VLN setup proposed in this work. The agent is required to return to its start location after arriving at the destination. The agent needs to remember its navigation histories to solve the task. We add a return command at the end of each instruction in R2R and a reverse path from the end to start locations as expert demonstration.
- **R2R-Last** is another VLN setup introduced in this work. It only contains the last sentence from the original R2R instructions describing the final destination. The agent, hence, is required to navigate to the goal without detailed guidance and depends on its past experiences.

Evaluation metrics. We adopt standard metrics [1], including (1) Trajectory Length (TL): the agent's navigated path in meters; (2) Navigation Error (NE): the average distance in meters between the

agent's final position and the target; (3) Success Rate (SR): the ratio of trajectories reaching the destination with a maximum error of 3 meters to the target; and (4) Success Rate normalized by the ratio between the length of the shortest path and the predicted path (SPL), which is more relevant than SR as it balances the navigation accuracy and efficiency. The above metrics measure whether the agent successfully arrives at the target in shortest distance, however, trajectories in R4R and R2R-Back datasets are no more the shortest path between two locations. Therefore, we further employ three metrics to measure the path fidelity between the predicted path and target path, including (5) Coverage weighted by Length Score (CLS) [3]; (6) the normalized Dynamic Time Warping (nDTW) [50]; and (7) the Success weighted by nDTW (SDTW).

Implementation details. For the HAMT model, we set $N_L = 9$ for language transformer, $N_h = 2$ for panoramic transformer in hierarchical history encoding, and $N_x = 4$ for cross-modal transformer. There are $K = 36$ image views in each panoramic observation. We use ViT-B/16 [43] for image encoding if not otherwise specified. In the training with proxy tasks, we randomly select proxy tasks for each mini-batch with predefined ratio. We train HAMT for 200k iterations with freezed vision transformer using learning rate of 5e-5 and batch size of 64 on 4 NVIDIA 2080Ti GPUs ($\sim$2 days). The whole HAMT model is trained end-to-end for 20k iterations on 20 NVIDIA V100 GPUs with learning rate of 5e-5 for the vision transformer and 1e-5 for the others ($\sim$20 hours). Annotations in R2R training set and augmented pairs from [17] are used for training. We use RL+IM with $\lambda = 0.2$ in Eq (3) by default to improve the navigation policy. The model is fine-tuned for 100k iterations with learning rate of 1e-5 and batch size of 8 on a single GPU. We fix the language and vision encoders in fine-tuning if not otherwise specified. The best model is selected according to performance on val unseen split. For fair comparison, the same augmented data in [5] is used for R2R and we balance the augmented data with original groundtruth during training, while no augmented data is used for other datasets. We use greedy search in inference following the single-run setting. Please see supplementary material for more details about datasets, metrics and implementation details.

4.2 Ablation studies

In this section, we evaluate each component in the HAMT model, including: hierarchical history encoding, end-to-end training with proxy tasks, and fine-tuning objectives.

How important is the history encoding for VLN? For fair comparison with the state-of-the-art recurrent architecture RecBERT [5], we use the same Resnet152 visual features and train all the models from scratch with RL+IM objectives to avoid the influence of different weight initialization. The models are optimized for 300k iterations end-to-end except for the visual feature. Table 2 compares different history encoding approaches on R2R dataset. Our recurrent model slightly differs from RecBERT (no init. OSCAR) [5] in transformer architecture as shown in Figure 1. It achieves slightly better performance on the validation unseen split. The temporal-only model uses the transformer to encode agent's oriented visual observations in the history, and outperforms the recurrent method by relative gains of 1.9% on SR and 2.1% on SPL for the unseen split. Adding panoramic observations in a hierarchical way results in 4.2% (SR) and 3.6% (SLP) relative improvements on the val unseen split compared to the recurrent method. Even lager improvements are achieved on the val seen split as the hierarchical model has a larger capacity to fit the seen environments. This evaluation demonstrates the advantage of our hierarchical history representation compared to the recurrent and temporal-only history representation.

Table 2: R2R navigation results for alternative methods of history encoding. All methods use Resnet152 visual features and are trained from scratch on the R2R dataset.

History Encoding	Val Seen		Val Unseen	
	SR↑	SPL↑	SR↑	SPL↑
RecBERT [5]	62	59	50	46
Recurrent	$60.9_{\pm 1.0}$	$56.6_{\pm 1.1}$	$52.2_{\pm 0.7}$	$47.0_{\pm 0.5}$
Temporal-only	$61.5_{\pm 0.8}$	$57.7_{\pm 0.7}$	$53.2_{\pm 0.1}$	$48.0_{\pm 0.4}$
Hierarchical	$\mathbf{65.5}_{\pm 1.2}$	$\mathbf{61.3}_{\pm 1.4}$	$\mathbf{54.4}_{\pm 0.4}$	$\mathbf{48.7}_{\pm 0.4}$

How much does training with proxy tasks help? We next evaluate the advantage of training HAMT end-to-end with proxy tasks. In Table 3a, the first row is directly trained with the RL+IM objective without initializing from weights trained by proxy tasks, while the second row uses proxy tasks for training prior to the RL+IM fine-tuning. We can see that it significantly boosts the performance to first train with proxy tasks. It improves on the unseen split with 16.7% and 18.0% relative gains on SR and SPL respectively. This indicates that training with auxiliary proxy tasks enables better generalization. Changing the visual feature from Resnet152 to ViT improves the

Table 3: Ablations for end-to-end HAMT training on R2R dataset using proposed proxy tasks.

(a) Comparison of visual features and end-to-end training. The "PT" stands for proxy tasks in training; "e2e" for optimizing the visual representation.

feature	PT	e2e	Val Seen SR↑	Val Seen SPL↑	Val Unseen SR↑	Val Unseen SPL↑
Resnet 152	×	×	$65.5_{\pm1.2}$	$61.3_{\pm1.4}$	$54.4_{\pm0.4}$	$48.7_{\pm0.4}$
	✓	×	$69.3_{\pm1.0}$	$64.8_{\pm1.2}$	$63.5_{\pm0.5}$	$57.5_{\pm0.5}$
ViT	✓	×	$\mathbf{75.7}_{\pm1.0}$	$\mathbf{72.5}_{\pm1.0}$	$64.4_{\pm0.3}$	$58.8_{\pm0.0}$
	✓	✓	$75.0_{\pm0.9}$	$71.7_{\pm0.7}$	$\mathbf{65.7}_{\pm0.7}$	$\mathbf{60.9}_{\pm0.7}$

(b) Comparison of different proxy tasks. The "SAP(R)" denotes the single step action prediction and regression task, and "SPREL" is the spatial relationship prediction task.

SAP (R)	SP REL	Val Seen SR↑	Val Seen SPL↑	Val Unseen SR↑	Val Unseen SPL↑
×	×	$71.2_{\pm2.3}$	$67.2_{\pm2.0}$	$62.8_{\pm1.3}$	$57.7_{\pm1.0}$
✓	×	$74.7_{\pm0.6}$	$71.1_{\pm0.9}$	$63.6_{\pm0.1}$	$58.1_{\pm0.4}$
✓	✓	$\mathbf{75.7}_{\pm1.0}$	$\mathbf{72.5}_{\pm1.0}$	$\mathbf{64.4}_{\pm0.3}$	$\mathbf{58.8}_{\pm0.0}$

performance, showing that more powerful visual representations matter. Finally, training the image encoder ViT end-to-end obtains additional gains. This is the first time to demonstrate that optimizing visual representations end-to-end is beneficial for the VLN task. In Table 3b, we evaluate the benefit of the two new proxy tasks for frozen ViT features using the other proxy tasks by default. The SAP(R) uses imitation learning to predict actions, which directly influences the navigation policy and improves the performance by a large margin. The SPREL is a self-supervised proxy task that forces the model to learn spatial relationships in panorama and helps generalization in unseen environments.

What is the impact of the fine-tuning objectives? Table 4 presents results using different objectives in fine-tuning. The first row directly applies HAMT trained by proxy tasks, which achieves lower performance than that after IM fine-tuning, because we mainly use augmented data in proxy task training to increase visual diversity, but such noisy data deteriorates action prediction performance. Previous work [22] has shown that RL alone performs poorly. However, our weights trained with proxy tasks stabilizes the RL fine-tuning. HAMT optimized by RL achieves much better performance than that when fine-tuning with IM on the SR metric. It indicates that RL is able to learn better exploration strategy on unseen environments. However, as the reward for RL focuses more on shortest path rather than path fidelity with instructions, the improvement on the SPL metric is relatively small compared to the SR metric. Moreover, the fluctuations of the pure RL objective is larger than IM. Therefore, mixing the RL and IM achieves the best performance.

Table 4: Ablations for fine-tuning objectives of sequential action prediction on R2R dataset.

IM	RL	Val Seen SR↑	Val Seen SPL↑	Val Unseen SR↑	Val Unseen SPL↑
×	×	57.9	54.8	51.8	48.9
✓	×	$63.7_{\pm2.1}$	$61.7_{\pm2.2}$	$57.2_{\pm0.1}$	$54.7_{\pm0.3}$
×	✓	$70.5_{\pm2.9}$	$65.6_{\pm2.8}$	$63.5_{\pm1.4}$	$57.5_{\pm1.1}$
✓	✓	$\mathbf{75.0}_{\pm0.9}$	$\mathbf{71.7}_{\pm0.7}$	$\mathbf{65.7}_{\pm0.7}$	$\mathbf{60.9}_{\pm0.7}$

4.3 Comparison to state of the art

Fine-grained VLN: R2R. Table 5 compares HAMT with previous VLN methods on the R2R benchmark. Our model outperforms state-of-the-art results of RecBERT [5] by relative 5.9% and 7.0% improvements in SPL on validation seen and unseen splits respectively. We also achieve state-of-the-art performance under the single-run setting on the unseen testing split of the leaderboard.[1] It demonstrates the effectiveness and generalization of our model.

Table 5: Comparison with state-of-the-art methods on R2R dataset.

Methods	Validation Seen				Validation Unseen				Test Unseen			
	TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑	TL	NE↓	SR↑	SPL↑
Seq2Seq [6]	11.33	6.01	39	-	8.39	7.81	22	-	8.13	7.85	20	18
SF [11]	-	3.36	66	-	-	6.62	35	-	14.82	6.62	35	28
PRESS [15]	10.57	4.39	58	55	10.36	5.28	49	45	10.77	5.49	49	45
EnvDrop [22]	11.00	3.99	62	59	10.70	5.22	52	48	11.66	5.23	51	47
AuxRN [51]	-	3.33	70	67	-	5.28	55	50	-	5.15	55	51
PREVALENT [17]	10.32	3.67	69	65	10.19	4.71	58	53	10.51	5.30	54	51
RelGraph [27]	10.13	3.47	67	65	9.99	4.73	57	53	10.29	4.75	55	52
RecBERT [5]	11.13	2.90	72	68	12.01	3.93	63	57	12.35	4.09	63	57
HAMT (Ours)	11.15	**2.51**	**76**	**72**	11.46	**2.29**	**66**	**61**	12.27	**3.93**	**65**	**60**

[1] We report the published results on the testing unseen split as shown in `https://eval.ai/web/challenges/challenge-page/97/leaderboard/270`.

Table 6: Comparison on R4R validation unseen split.

Methods	NE↓	SR↑	CLS↑	nDTW↑	SDTW↑
SF [11]	8.47	24	30	-	-
RCM [28]	-	29	35	30	13
PTA [32]	8.25	24	37	32	10
EGP [13]	8.0	30.2	44.4	37.4	17.5
RelGraph [27]	7.43	36	41	47	34
RecBERT† [5]	6.67	43.6	51.4	45.1	29.9
HAMT (Ours)	**6.09**	**44.6**	**57.7**	**50.3**	**31.8**

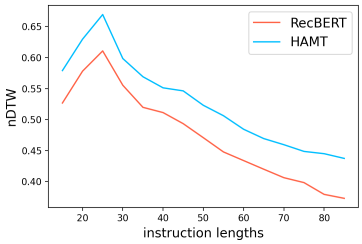


Figure 3: nDTW with respect to instruction length on R4R val unseen split.

Long-horizon VLN: R4R and R2R-Back. Table 6 shows navigation results for R4R dataset, which contains longer instructions and trajectories compared to R2R. Therefore, we use the encoder-decoder variant of HAMT on R4R dataset. Our method outperforms previous approaches in all metrics and shows particularly large improvements for the path fidelity related metrics. Compared to RecBERT, HAMT achives 8.2% and 9.5% relative improvement in CLS and nDTW respectively. The large improvements on these path fidelity related metrics indicate that HAMT is better to follow the designated path of the fine-grained instruction. Figure 3 evaluates the performance of HAMT and RecBERT with respect to instruction length measured by words. Though the nDTW decreases for longer instructions, the relative improvement of HAMT increases with the instruction length.

The navigation performance on R2R-Back dataset is presented in Table 7. We compare with two state-of-the-art recurrent models EnvDrop [22] and RecBERT [5] based on LSTM and transformer respectively. The improvements are more significant on this task as it requires the agent to remember the way it came to the target to successfully return back. The recurrent state is insufficient to capture such history and leads to inferior performance compared to the HAMT model.

Table 7: Comparison of methods on the R2R-Back dataset.

Methods	Val Seen					Val Unseen				
	TL	SR↑	SPL↑	nDTW↑	SDTW↑	TL	SR↑	SPL↑	nDTW↑	SDTW↑
EnvDrop† [22]	23.83	44.1	42.0	61.3	39.4	24.57	32.4	30.2	51.1	28.0
RecBERT† [5]	22.33	51.4	48.4	67.3	45.7	23.35	41.1	37.7	58.2	35.6
HAMT (Ours)	22.76	**64.8**	**61.8**	**73.7**	**58.9**	23.78	**57.2**	**53.1**	**65.1**	**49.5**

VLN with high-level instructions: R2R-Last. Table 8 shows results on the R2R-Last dataset that specifies the goal location and contains no step-by-step instructions. The HAMT model with the hierarchical history encoding is able to better accumulate the knowledge of the environment and achieves 9.8% and 10.5% relative gains on SPL metric on seen and unseen splits respectively compared to RecBERT [5].

Table 8: Comparison on the R2R-Last dataset.

Methods	Val Seen		Val Unseen	
	SR↑	SPL↑	SR↑	SPL↑
EnvDrop† [22]	42.8	38.4	34.3	28.3
RecBERT† [5]	50.2	45.8	41.6	37.3
HAMT (Ours)	**53.3**	**50.3**	**45.2**	**41.2**

5 Conclusion

This paper presents the first end-to-end transformer for vision-and-language navigation, denoted as History Aware Multimodal Transformer (HAMT). Our method efficiently encodes long-horizon history and combines it with instructions and observations to derive multimodal action prediction. The HAMT is first trained with proxy tasks in an end-to-end manner, and is then fine-tuned with RL to improve the navigation policy. We achieve state-of-the-art navigation performance on three challenging VLN tasks, demonstrating improved accuracy and generalization of our approach compared to the dominant recurrent methods. Future work could extend our history-aware transformer to VLN with continuous actions [52] and could benefit from pretraining on large-scale navigation datasets such as StreetView [53]. This paper has minimal ethical, privacy and safety concerns.

† We run the original implementation of methods released by the authors.

References

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

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

[3] Vihan Jain, Gabriel Magalhaes, Alexander Ku, Ashish Vaswani, Eugene Ie, and Jason Baldridge. Stay on the path: Instruction fidelity in vision-and-language navigation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1862–1872, 2019.

[4] Yubo Zhang, Hao Tan, and Mohit Bansal. Diagnosing the environment bias in vision-and-language navigation. *International Joint Conferences on Artificial Intelligence*, 2020.

[5] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. A recurrent vision-and-language BERT for navigation. *arXiv preprint arXiv:2011.13922*, 2020.

[6] 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*, pages 3674–3683, 2018.

[7] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-across-room: Multilingual vision-and-language navigation with dense spatiotemporal grounding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4392–4412, 2020.

[8] Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. Reverie: Remote embodied visual referring expression in real indoor environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9982–9991, 2020.

[9] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR, 2020.

[10] Samyak Datta, Karan Sikka, Anirban Roy, Karuna Ahuja, Devi Parikh, and Ajay Divakaran. Align2ground: Weakly supervised phrase grounding guided by image-caption alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2601–2610, 2019.

[11] Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. Speaker-follower models for vision-and-language navigation. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 3318–3329, 2018.

[12] Kuan Fang, Alexander Toshev, Li Fei-Fei, and Silvio Savarese. Scene memory transformer for embodied agents in long-horizon tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 538–547, 2019.

[13] Zhiwei Deng, Karthik Narasimhan, and Olga Russakovsky. Evolving graphical planner: Contextual global planning for vision-and-language navigation. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[14] Hanqing Wang, Wenguan Wang, Wei Liang, Caiming Xiong, and Jianbing Shen. Structured scene memory for vision-language navigation. *arXiv preprint arXiv:2103.03454*, 2021.

[15] Xiujun Li, Chunyuan Li, Qiaolin Xia, Yonatan Bisk, Asli Celikyilmaz, Jianfeng Gao, Noah A Smith, and Yejin Choi. Robust navigation with language pretraining and stochastic sampling. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1494–1499, 2019.

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[17] Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. Towards learning a generic agent for vision-and-language navigation via pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13137–13146, 2020.

[18] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *International Conference on Learning Representations*, 2016.

[19] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[20] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, volume 28, 2015.

[21] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

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

[23] Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, et al. Stabilizing transformers for reinforcement learning. In *International Conference on Machine Learning*, pages 7487–7498. PMLR, 2020.

[24] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[25] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. Alfred: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749, 2020.

[26] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. *arXiv preprint arXiv:1901.03035*, 2019.

[27] Yicong Hong, Cristian Rodriguez-Opazo, Yuankai Qi, Qi Wu, and Stephen Gould. Language and visual entity relationship graph for agent navigation. *arXiv preprint arXiv:2010.09304*, 2020.

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

[29] Hu Wang, Qi Wu, and Chunhua Shen. Soft expert reward learning for vision-and-language navigation. *arXiv preprint arXiv:2007.10835*, 2020.

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

[31] Xiangru Lin, Guanbin Li, and Yizhou Yu. Scene-intuitive agent for remote embodied visual grounding. *arXiv preprint arXiv:2103.12944*, 2021.

[32] Federico Landi, Lorenzo Baraldi, Marcella Cornia, Massimiliano Corsini, and Rita Cucchiara. Perceive, transform, and act: Multi-modal attention networks for vision-and-language navigation. *arXiv preprint arXiv:1911.12377*, 2019.

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

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

[35] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2017.

[36] Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. *International Conference on Learning Representations*, 2018.

[37] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic transformer for vision-and-language navigation, 2021.

[38] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *European Conference on Computer Vision*, pages 104–120. Springer, 2020.

[39] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer, 2020.

[40] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[41] Hao Tan and Mohit Bansal. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5103–5114, 2019.

[42] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. *arXiv preprint arXiv:2102.03334*, 2021.

[43] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16$\times$ 16 words: Transformers for image recognition at scale. *International Conference on Learning Representations*, 2020.

[44] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *European Conference on Computer Vision*, pages 259–274. Springer, 2020.

[45] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[46] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. ViVit: A video vision transformer. *arXiv preprint arXiv:2103.15691*, 2021.

[47] Jize Cao, Zhe Gan, Yu Cheng, Licheng Yu, Yen-Chun Chen, and Jingjing Liu. Behind the scene: Revealing the secrets of pre-trained vision-and-language models. In *European Conference on Computer Vision*, pages 565–580. Springer, 2020.

[48] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. IEEE, 2009.

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

[50] Gabriel Ilharco, Vihan Jain, Alexander Ku, Eugene Ie, and Jason Baldridge. General evaluation for instruction conditioned navigation using dynamic time warping. *arXiv preprint arXiv:1907.05446*, 2019.

[51] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10012–10022, 2020.

[52] Jacob Krantz, Erik Wijmans, Arjun Majumdar, Dhruv Batra, and Stefan Lee. Beyond the nav-graph: Vision-and-language navigation in continuous environments. In *European Conference on Computer Vision*, pages 104–120. Springer, 2020.

[53] Piotr Mirowski, Matthew Koichi 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 *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 2424–2435, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Conclusion.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Conclusion.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We included codes and instructions in the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] The training details are presented in Section 4 and the supplementary material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 4
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] The computation costs are presented in Section 4 and the supplementary material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]