Episodic Transformer for Vision-and-Language Navigation

Alexander Pashevich[1*] Cordelia Schmid[2] Chen Sun[2,3]
[1] Inria [2] Google Research [3] Brown University

Abstract

Interaction and navigation defined by natural language instructions in dynamic environments pose significant challenges for neural agents. This paper focuses on addressing two challenges: handling long sequence of subtasks, and understanding complex human instructions. We propose Episodic Transformer (E.T.), a multimodal transformer that encodes language inputs and the full episode history of visual observations and actions. To improve training, we leverage synthetic instructions as an intermediate representation that decouples understanding the visual appearance of an environment from the variations of natural language instructions. We demonstrate that encoding the history with a transformer is critical to solve compositional tasks, and that pretraining and joint training with synthetic instructions further improve the performance. Our approach sets a new state of the art on the challenging ALFRED benchmark, achieving 38.4% *and* 8.5% *task success rates on seen and unseen test splits.*

1. Introduction

Having an autonomous agent performing various household tasks is a long-standing goal of the research community. To benchmark research progress, several simulated environments [3, 53, 56] have recently emerged where the agents navigate and interact with the environment following natural language instructions. Solving the vision-and-language navigation (VLN) task requires the agent to ground human instructions in its embodied perception and action space. In practice, the agent is often required to perform long compositional tasks while observing only a small fraction of the environment from an egocentric point of view. Demonstrations manually annotated with human instructions are commonly used to teach an agent to accomplish specified tasks.

This paper attempts to address two main challenges of VLN: (1) handling highly compositional tasks consisting of many subtasks and actions; (2) understanding the complex human instructions that are used to specify a task. Figure 1 shows an example task that illustrates both challenges. We show six key steps from a demonstration of 53 actions. To fulfill the task, the agent is expected to remember the location of a *fireplace* at $t = 0$ and use this knowledge much later (at $t = 31$). It also needs to solve object- (*e.g.* "another vase") and location-grounded (*e.g.* "where you were standing previously") coreference resolution in order to understand the human instructions.

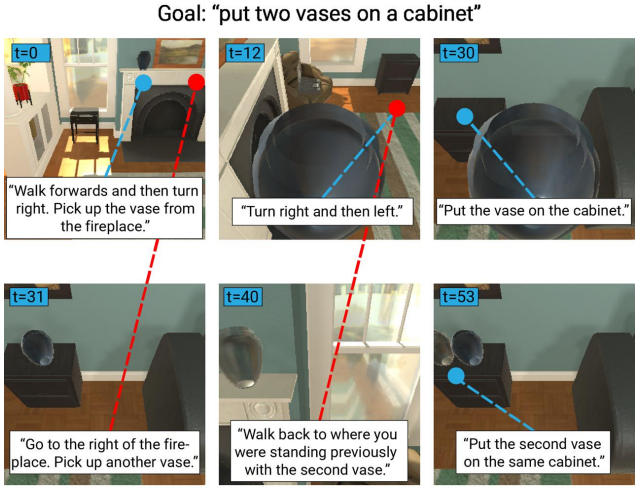

Figure 1: An example of a compositional task in the ALFRED dataset [56] where the agent is asked to bring two vases to a cabinet. We show several frames from an expert demonstration with corresponding step-by-step instructions. The instructions expect the agent to be able to navigate to a *fireplace* which is not visible in its current egocentric view and to remember its previous location by referring to it as *"where you were standing previously"*.

Addressing the first challenge requires the agent to remember its past actions and observations. Most recent VLN approaches rely on recurrent architectures [39, 60, 68, 73] where the internal state is expected to keep information about previous actions and observations. However, the recurrent networks are known to be inefficient in capturing long-term dependencies [66] and may fail to execute long action sequences [25, 56]. Motivated by the success of the attention-based transformer architecture [65] at language understanding [9, 17] and multimodal learning [18, 59], we propose to use a transformer encoder to combine multi-

*Work done as an intern at Google Research.

modal inputs including camera observations, language instructions, and previous actions. The transformer encoder has access to the history of the *entire episode* to allow long-term memory and outputs the action to take next. We name our proposed architecture **Episodic Transformer (E.T.)**.

Addressing the second challenge requires revisiting different ways to specify a task for the autonomous agent. We observe that domain-specific language [22] and temporal logic [24, 43] can unambiguously specify the target states and (optionally) their temporal dependencies, while being decoupled from the visual appearance of a certain environment and the variations of human instructions. We hypothesize that using these *synthetic instructions* as an intermediate interface between the human and the agent would help the model to learn more easily and generalize better. To this end, we propose to pretrain the transformer-based language encoder in E.T. by predicting the synthetic instructions from human instructions. We also explore joint training, where human instructions and synthetic instructions are mapped into a shared latent space.

To evaluate the performance of E.T., we use the ALFRED dataset [56] which consists of longer episodes than the other vision-and-language navigation datasets [3, 13, 53] and also requires object interaction. We experimentally show that E.T. benefits from full episode memory and is better at solving tasks with long horizons than recurrent models. We also observe significant gains by pretraining the language encoder with the synthetic instructions. Furthermore, we show that when used for training jointly with natural language such intermediate representations outperform conventional data augmentation techniques for vision-and-language navigation [20] and work better than image-based annotations [37].

In summary, our two main contributions are as follows. First, we propose Episodic Transformer (E.T.), an attention-based architecture for vision-and-language navigation, and demonstrate its advantages over recurrent models. Second, we propose to use synthetic instructions as the intermediate interface between the human and the agent. Both contributions combined allow us to achieve a new state-of-the-art on the challenging ALFRED dataset.

Code and models are available on the project page[1].

2. Related work

Instruction following agents. Building systems to understand and execute human instructions has been the subject of many previous works [7, 8, 10, 12, 37, 41, 46, 47, 52, 62]. Instruction types include structured commands or logic programs [22, 43, 53], natural language [12, 61], target state images [37], or a mix [38]. While earlier work focuses on mapping instructions and structured world states into actions [4, 45, 51], it is desirable for the agents to be able to handle raw sensory inputs, such as images or videos. To address this, the visual-and-language navigation (VLN) task is proposed to introduce rich and unstructured visual context for the agent to explore, perceive and execute upon [3, 13, 32, 33, 44]. The agent is requested to navigate to the target location based on human instructions and real, or photo-realistic image inputs, implemented as navigation graphs [3, 13] or a continuous environment [32] in simulators [16, 31, 55, 63]. More recently, the ALFRED environment [56] introduces the object interaction component to complement visual-language navigation. It is a more challenging setup as sequences are longer than in other vision-language navigation datasets and all steps of a sequence have to be executed properly to succeed. We focus on the ALFRED environment and its defined tasks.

Training a neural agent for VLN. State-of-the-art models in language grounded navigation are neural agents trained using either Imitation Learning [20], Reinforcement Learning [34], or a combination of both [60, 68]. In addition, auxiliary tasks, such as progress estimation [39, 40], backtracking [30], speaker-driven route selection [20], cross-modal matching [29, 68], back translation [60], pretraining on subtasks [74], and text-based pretraining [14, 57] are proposed to improve the performance and generalization of neural agents in seen and unseen environments. Most of these approaches use recurrent neural networks and encode previous observations and actions as hidden states. Our work proposes to leverage transformers [65] which enables encoding the full episode of history for long-term navigation and interaction. Most relevant to our approach are VLN-BERT [42] and Recurrent VLBERT [28], which also employ transformers for VLN. Unlike our approach, VLN-BERT [42] trains a transformer to measure the compatibility of an instruction and a set of already generated trajectories. Concurrently, Recurrent VLBERT [28] uses an explicit recurrent state and a pretrained VLBERT to process one observation for each timestep, which might have difficulty solving long-horizon tasks [66] such as ALFRED. In contrast, we do not introduce any recurrency and process all the history of observations at once.

Multimodal Transformers. Transformers [65] have brought success to a wide range of classification and generation tasks, from language [9, 17, 65] to images [11, 19] and videos [23, 67]. In [48], the authors show that training transformers for long time horizon planning with RL is challenging and propose a solution. The convergence of the transformer architecture for different problem domains also leads to multimodal transformers, where a unified transformer model is tasked to solve problems that require multimodal information, such as visual question answering [36], video captioning and temporal prediction [59], or retrieval [21]. Our Episodic Transformer can be considered

[1] https://github.com/alexpashevich/E.T.

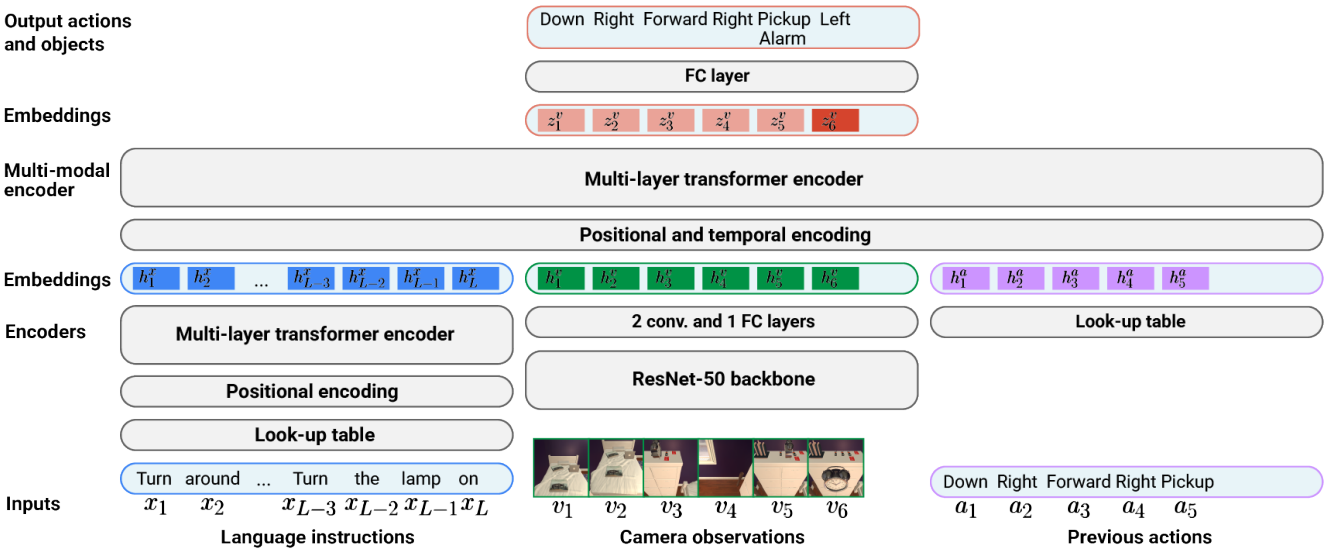

Figure 2: Episodic Transformer (E.T.) architecture. To predict the next action, the E.T. model is given a natural language instruction $x_{1:L}$, visual observations since the beginning of an episode $v_{1:t}$, and previously taken actions $a_{1:t-1}$. Here we show an example that corresponds to the 6$^{\text{th}}$ timestep of an episode: $t = 6$. After processing $x_{1:L}$ with a transformer-based language encoder, embedding $v_{1:t}$ with a ResNet-50 backbone and passing $a_{1:t-1}$ through a look-up table, the agent outputs t actions. During training we use all predicted actions for a gradient descent step. At test time, we apply the last action a_t to the environment.

a multimodal transformer, where the inputs are language (instructions), vision (images), and actions.

Semantic parsing of human instructions. Semantic parsing focuses on converting natural language into logic forms that can be interpreted by machines. It has applications in question answering [6,70,71] and can be learned either with paired supervision [6, 69, 72] or weak supervision [5, 50]. For instruction following, semantic parsing has been applied to map natural language into lambda calculus expressions [5] or linear temporal logic [50]. We show that rather than directly using the semantic parsing outputs, it is more beneficial to transfer its pretrained language encoder to the downstream VLN task.

3. Method

We first define the vision-and-language navigation task in Section 3.1 and describe the Episodic Transformer (E.T.) model in Section 3.2. We then introduce the synthetic language and explain how we leverage it for pretraining and joint training in Section 3.3.

3.1. VLN background

The vision-and-language navigation task requires an agent to navigate in an environment and to reach a goal specified by a natural language instruction. Each demonstration is a tuple $(x_{1:L}, v_{1:T}, a_{1:T})$ of a natural language instruction, expert visual observations, and expert actions. The instruction $x_{1:L}$ is a sequence of L word tokens $x_i \in \mathbb{R}$. The visual observations $v_{1:T}$ is a sequence of T camera images $v_t \in \mathbb{R}^{W \times H \times 3}$ where T is the demonstration length and $W \times H$ is the image size. The expert actions $a_{1:T}$ is a sequence of T action type labels $a_t \in \{1, \ldots, A\}$ used by the expert and A is the number of action types.

The goal is to learn an agent function f that approximates the expert policy. In the case of a recurrent architecture, the agent predicts the next action $\hat{a}_t$ given a language instruction $x_{1:L}$, a visual observation v_t, the previously taken action $\hat{a}_{t-1}$, and uses its hidden state h_{t-1} to keep track of the history:

$$\hat{a}_t, h_t = f(x_{1:L}, v_t, \hat{a}_{t-1}, h_{t-1}). \quad (1)$$

For an agent with full episode observability, all previous visual observations $v_{1:t}$ and all previous actions $\hat{a}_{1:t-1}$ are provided to the agent directly and no hidden state is required:

$$\hat{a}_t = f(x_{1:L}, v_{1:t}, \hat{a}_{1:t-1}). \quad (2)$$

3.2. Episodic Transformer model

Our Episodic Transformer (E.T.) model shown in Figure 2 relies on attention-based multi-layer transformer encoders [65]. It has no hidden state and observes the full history of visual observations and previous actions. To inject information about the order of words, frames, and action sequences, we apply the sinusoidal encoding to transformer inputs. We refer to this encoding as positional encoding for language tokens and temporal encoding for expert observations and actions.

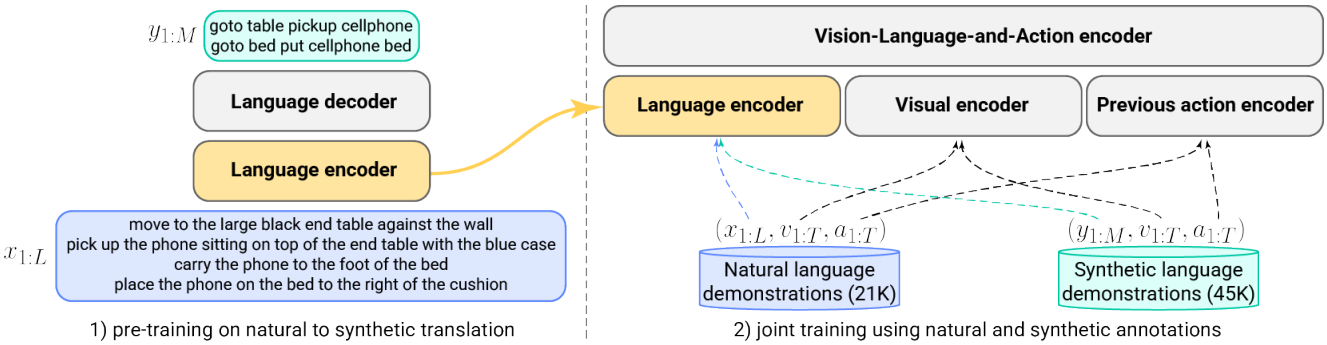

Figure 3: Training with natural and synthetic language. Left: We pretrain the language encoder of the model to translate natural language instructions to synthetic language. Due to a more task-oriented synthetic representation, the language encoder learns better representations. We use the language encoder weights to initialize the language encoder of the agent (shown in yellow). Right: We jointly use demonstrations annotated with natural language and demonstrations annotated with synthetic language to train the agent. Due to the larger size of the synthetic language dataset, the resulting agent has better performance even when evaluated on natural language annotations.

Our E.T. architecture consists of four encoders: language encoder, visual encoder, action encoder, and multimodal encoder. The language encoder shown in the bottom-left part of Figure 2 gets instruction tokens $x_{1:L}$ as input. It consists of a look-up table and a multi-layer transformer encoder and outputs a sequence of contextualized language embeddings $h^x_{1:L}$. The visual encoder shown in the bottom-center part of Figure 2 is a ResNet-50 backbone [27] followed by 2 convolutional and 1 fully-connected layers. The visual encoder projects a visual observation v_t into its embedding h^v_t. All the episode visual observations $v_{1:T}$ are projected independently using the same encoder. The action encoder is a look-up table shown in the bottom-right part of Figure 2 which maps action types $a_{1:T}$ into action embeddings $h^a_{1:T}$.

The multimodal encoder is a multi-layer transformer encoder shown in the middle of Figure 2. Given the concatenated embeddings from modality-specific encoders $(h^x_{1:L}, h^v_{1:T}, h^a_{1:T})$, the multimodal encoder returns output embeddings $(z^x_{1:L}, z^v_{1:T}, z^a_{1:T})$. The multimodal encoder employs causal attention [65] to prevent visual and action embeddings from attending to subsequent timesteps. We take the output embeddings $z^v_{1:T}$ and add a single fully-connected layer to predict agent actions $\hat{a}_{1:T}$.

During E.T. training, we take advantage of the sequential nature of the transformer architecture. We input a language instruction $x_{1:L}$ as well as all visual observations $v_{1:T}$ and all actions $a_{1:T}$ of an expert demonstration to the model. The E.T. model predicts all actions $\hat{a}_{1:T}$ at once as shown at the top of Figure 2. We compute and minimize the cross-entropy loss between predicted actions $\hat{a}_{1:T}$ and expert actions $a_{1:T}$. During testing at timestep t, we input visual observations $v_{1:t}$ up to a current timestep and previous actions $\hat{a}_{1:t-1}$ taken by the agent. We select the action predicted for the last timestep $\hat{a}_t, \hat{c}_t$ and apply it to the environment which generates the next visual observation v_{t+1}. In Figure 2 we show an example that corresponds to the 6$^{\text{th}}$ timestep of an episode where the action `Left` will be taken next.

3.3. Synthetic language

To improve understanding of human instructions that present a wide range of variability, we propose to pretrain the agent language encoder with a translation into a synthetic language, see Figure 3 (left). We also generate additional demonstrations, annotate them with synthetic language and jointly train the agent using both synthetic and natural language demonstrations, see Figure 3 (right).

An example of the synthetic language and a corresponding natural language instruction is shown in Figure 3 (left). The synthetic annotation is generated for each expert demonstration using the expert path planner arguments. In ALFRED, each expert path is defined with Planning Domain Definition Language (PDDL) [22] which consists of several subgoal actions. Each subgoal action has a type and a target class, e.g. `Put Apple Table` or `Goto Bed` which we use as a synthetic annotation for this subgoal action. Note that such annotation only defines a class but not an instance of the target. We annotate each expert demonstration with subgoal action annotations concatenated in chronological order to produce a synthetic annotation $y_{1:M}$.

We use synthetic language to pretrain the language encoder of the agent on a sequence-to-sequence (seq2seq) translation task. The translation dataset consists of corresponding pairs $(x_{1:L}, y_{1:M})$ of natural and synthetic instructions. The translation model consists of a language encoder and a language decoder as shown in Figure 3 (left). The language encoder is identical to the agent language encoder described in Section 3.2. The language decoder is a multi-layer transformer decoder with positional encoding and the same hyperparameters as the encoder. Given a natural language annotation $x_{1:L}$, we use the language encoder to produce embeddings $h_{1:L}$. The embeddings are passed to the language decoder which predicts N translation tokens $\hat{y}_i$. We train the model by minimizing the cross-entropy loss between predictions $\hat{y}_{1:N}$ and synthetic annotations $y_{1:M}$.

Once the training converges, we use the weights of the translator language encoder to initialize the language encoder of the agent.

We also explore joint training by generating an additional dataset of expert demonstrations annotated with synthetic language. We use the AI2-THOR simulator [31] and scripts provided by Shridhar *et al*. [56]. Apart from the annotations, the synthetic dataset differs from the original one in terms of objects configurations and agent initial positions. We train the agent to predict actions using both natural and synthetic language datasets as shown on the right in Figure 3. We use the same language, vision, and action encoders for both datasets but two different look-up tables for natural and synthetic language tokens which we found to work the best experimentally. For both datasets, we sample batches of the same size, compute the two losses and do a single gradient descent step. After a fixed number of training epochs, we evaluate the agent on natural and synthetic language separately using the same set of validation tasks.

4. Results

In this section, we ablate different components of E.T. and compare E.T. with state-of-the-art methods. First, we describe the experimental setup and the dataset in Section 4.1. Next, we compare our method to a recurrent baseline and highlight the importance of full episode observability in Section 4.2. We then study the impact of joint training and pretraining with synthetic instructions in Section 4.3 and compare with previous state-of-the-art methods on the ALFRED dataset in Section 4.4.

4.1. Experimental setup

Dataset. The ALFRED dataset [56] consists of demonstrations of an agent performing household tasks following goals defined with natural language. The tasks are compositional with nonreversible state changes. The dataset includes $8,055$ expert trajectories $(v_{1:T}, a_{1:T})$ annotated with $25,743$ natural language instructions $x_{1:L}$. It is split into $21,023$ train, $1,641$ validation, and $3,062$ test annotations. The validation and test folds are divided into *seen* splits which contain environments from the train fold and *unseen* splits which contain new environments. To leverage synthetic instructions to pretrain a language encoder, we pair every annotated instruction $x_{1:L}$ with its corresponding synthetic instruction $y_{1:M}$ in the train fold. For joint training, we generate $44,996$ demonstrations $(y_{1:M}, v_{1:T}, a_{1:T})$ from the train environments annotated automatically with synthetic instructions. For ablation studies in Section 4.2 and Section 4.3, we use the validation folds only. For comparison with state-of-the-art in Section 4.4, we report results on both validation and test folds.

Baselines. In Section 4.2, we compare our model to a model based on a bi-directional LSTM [56]. We use the same hyperparameters as Shridhar *et al*. [56] and set the language encoder hidden size to 100, the action decoder hidden size to 512, the visual embeddings size to 2500, and use 0.3 dropout for the decoder hidden state. We experimentally find the Adam optimizer with no weight decay and a weight coefficient 0.1 for the target class cross-entropy loss to work best. The LSTM model uses the same visual encoder as the E.T. model. In Section 4.4, we also compare our model to MOCA [58] and the model of Nguyen *et al*. [64].

Evaluation metrics. For our ablation studies in Sections 4.2 and 4.3, we report agent success rates. To understand the performance difference with recurrent-based architectures in Section 4.2, we also report success rates on individual subgoals. This metric corresponds to the proportion of subgoal tasks completed after following an expert demonstration until the beginning of the subgoal and conditioned on the entire language instruction. We note that the average task length is 50 timesteps while the average length of a subgoal is 7 timesteps.

Implementation details. Among the 13 possible action types, 7 actions involve interacting with a target object in the environment. The target object of an action a_t is chosen with a binary mask $m_t \in \{0,1\}^{W \times H}$ that specifies the pixels of visual observation v_t that belong to the target object. There are 119 object classes in total. The pixel masks m_t are provided along with expert demonstrations during training. We follow Singh *et al*. [58] and ask our agent to predict the target object class c_t, which is then used to retrieve the corresponding pixel mask $\hat{m}_t$ generated by a pretrained instance segmentation model. The segmentation model takes v_t as input and outputs $(\hat{c}_t, \hat{m}_t)$.

The agent observations are resized to 224×224. The mask generator receives images of size 300×300 following Singh *et al*. [58]. Both the visual encoder and the mask generator are pretrained on a dataset of 325K frames of expert demonstrations from the train fold and corresponding class segmentation masks. We use ResNet-50 Faster R-CNN [54] for the visual encoder pretraining and ResNet-50 Mask R-CNN [26] for the mask generator. We do not update the mask generator and the visual encoder ResNet backbone during the agent training. In the visual encoder, ResNet features are average-pooled 4 times to reduce their size and 0.3 dropout is applied. Resulting feature maps of $512 \times 7 \times 7$ are fed into 2 convolutional layers with 256 and 64 filters of size 1 by 1 and mapped into an embedding of the size 768 with a fully connected layer. Both transformer encoders of E.T. have 2 blocks, 12 self-attention heads, and the hidden size of 768. We use 0.1 dropout inside transformer encoders.

We use the AdamW optimizer [35] with 0.33 weight decay and train the model for 20 epochs. Every epoch includes $3,750$ batches of 8 demonstrations each. For joint training, each batch consists of 4 demonstrations with human instructions and 4 demonstrations with synthetic in-

Model	Task		Sub-goal	
	Seen	Unseen	Seen	Unseen
LSTM	23.2	2.4	75.5	58.7
LSTM + E.T. enc.	27.8	**3.3**	76.6	59.5
E.T.	**33.8**	3.2	**77.3**	**59.6**

Table 1: Comparison of E.T. and LSTM architectures: (1) an LSTM-based model [56], (2) an LSTM-based model trained with the transformer language encoder of the E.T. model, (3) E.T., our transformer-based model. All models are trained using the natural language dataset only and evaluated on validation folds. The two parts of the table show the success rate for tasks (average length 50) and sub-goals (average length 7). While the sub-goal success rates of all models are relatively close, E.T. outperforms both recurrent agents on full tasks which highlights the importance of the full episode observability.

Visible	Frames		Actions	
	Seen	Unseen	Seen	Unseen
None	0.5	0.2	23.7	1.7
1 last	28.9	2.2	**33.8**	**3.2**
4 last	31.5	2.0	32.0	2.4
16 last	33.5	2.9	31.1	2.8
All	**33.8**	**3.2**	27.1	2.2

Table 2: Ablation on accessible history length of E.T., in terms of visual frames (left two columns) and actions (right two columns).

structions. For all experiments, we use a learning rate of 10^{-4} during the first 10 epochs and 10^{-5} during the last 10 epochs. Following Shridhar *et al*. [56], we use auxiliary losses for overall and subgoal progress [39] which we sum to the model cross-entropy loss with weights 0.1. All the hyperparameter choices were made using a moderate size grid search. Once the training is finished, we evaluate every 2-nd epoch on the validation folds. Following Singh *et al*. [58], we use Instance Association in Time and Obstruction Detection modules during evaluation.

4.2. Model analysis

Comparison with recurrent models. To validate the gain due to the episodic memory, we compare the E.T. architecture with a model based on a recurrent LSTM architecture. We train both models using the dataset with natural language annotations only. As shown in Table 1, the recurrent model succeeds in 23.2% of tasks in seen environments and in 2.4% of tasks in unseen environments. E.T. succeeds in 33.8% and 3.2% of tasks respectively which is a relative improvement of 45.6% and 33.3% compared to the LSTM-based agent. However, the success rate computed for individual subgoals shows only 2.3% and 1.5% of relative improvement of E.T. over the recurrent agent in seen and unseen environments respectively. We note that a task consists on average of 6.5 subgoals which makes the long-term memory much more important for solving full tasks.

To understand the performance difference, we train an LSTM-based model with the E.T. language encoder. Given that both LSTM and E.T. agents receive the same visual features processed by the frozen ResNet-50 backbone and have the same language encoder architecture, the principal difference between the two models is the processing of previous observations. While the E.T. agent observes all previous frames using the attention mechanism, the LSTM-based model relies on its recurrent state and explicitly observes only the last visual frame. The recurrent model performance shown in the 2-nd row of Table 1 is similar to the E.T. performance in unseen environments but is 17.7% less successful than E.T. in seen environments. This comparison highlights the importance of the attention mechanism and full episode observability. We note that E.T. needs only one forward pass for a gradient descent update on a full episode. In contrast, the LSTM models need to do a separate forward pass for each episode timestep which significantly increases their training time with respect to E.T. models. We further compare how E.T. and LSTM models scale with additional demonstrations in Section 4.3.

Accessible history length. We train E.T. using different lengths of the episode history observed by the agent in terms of visual frames and previous actions and show the results in Table 2. The first two columns of Table 2 compare different lengths of visual observations history from no past frames to the entire episode. The results indicate that having access to all visual observations is important for the model performance. We note that the performance of the model with 16 input frames is close to the performance of the full episode memory agent, which can be explained by the average task length of 50 timesteps.

The last two columns of Table 2 show that the agent does not benefit from accessing more than one past action. This behavior can be explained by the "causal misidentification" phenomenon: access to more information can yield worse performance [15]. It can also be explained by poor generalizability due to the overfitting of the model to expert demonstrations. We also note that the model observing no previous actions is 29.8% and 46.8% relatively less successful in seen and unseen environments than the agent observing the last action. We, therefore, fix the memory size to be unlimited for visual observations and to be 1 timestep for the previous actions.

Model capacity. Transformer-based models are known to be expressive but prone to overfitting. We study how the model capacity impacts the performance while training on the original ALFRED dataset. We change the number of transformer blocks in the language encoder and the multimodal encoder and report results in Table 3. The results indicate that the model with a single transformer block is

# Blocks	Seen	Unseen
1	25.0	1.6
2	**33.8**	**3.2**
3	28.6	2.2
4	19.8	1.1

Table 3: Ablation of E.T. model capacity. We compare E.T. models with different number of transformer blocks in language and multimodal encoders.

Synthetic instr.	Test on synthetic		Test on human	
	Seen	Unseen	Seen	Unseen
Expert frames	**54.0**	**6.1**	28.5	3.4
Speaker text	36.3	3.1	37.4	3.9
Subgoal actions	47.2	5.9	**38.5**	**5.4**
No synthetic	-	-	33.8	3.2

Table 4: Comparison of different synthetic instructions used for **joint training**. We jointly train E.T. using demonstrations with human annotations and demonstrations with different types of synthetic instructions. In the first two columns, we evaluate the resulting models using the same type of synthetic annotations that is used during training. In the last two columns, the models are evaluated on human annotated instructions.

Train data	LSTM		E.T.	
	Seen	Unseen	Seen	Unseen
Human annotations	23.2	2.4	33.8	3.2
Human + synthetic	25.2	2.9	**38.5**	**5.4**

Table 5: Comparison of an LSTM-based model and E.T. **trained jointly with demonstrations annotated by subgoal actions**. The results indicate that E.T. scales better with additional data than the LSTM-based agent.

not expressive enough and the models with 3 and 4 blocks overfit to the train data. The model with 2 blocks represents a trade-off between under- and overfitting and we, therefore, keep this value for all the experiments.

Attention visualization. We visualize text and visual attention heatmaps in Appendices A.4 and A.5 of [49].

4.3. Training with synthetic annotations

Joint training. We train the E.T. model using the original dataset of $21,023$ expert demonstrations annotated with natural language and the additionally generated dataset of $44,996$ expert demonstrations with synthetic annotations. We compare three types of synthetic annotations: (1) direct use of visual embeddings from the expert demonstration frames, no language instruction is generated. A similar approach can be found in Lynch and Sermanet [38]; (2) train a model to generate instructions, *e.g.* with a speaker model [20], where the inputs are visual embeddings from the expert demonstration frames, and the targets are human-annotated instructions; and (3) subgoal actions and objects annotations described in Section 3.3. For (1), we experimentally find using all expert frames from a demonstration works significantly better than a subset of frames. The visual embeddings used in (1) and (2) are extracted from a pretrained frozen ResNet-50 described in Section 4.1. To generate speaker annotations, we use a transformer-based seq2seq model (Section 3.3) with the difference that the inputs are visual embeddings instead of text.

We report success rates of models trained jointly and evaluated independently on synthetic and human-annotated instructions in Table 4. The results are reported on the validation folds. The model trained on expert frames achieves the highest performance when evaluated on synthetic instructions. However, when evaluated on human instructions, this model has 15.6% relatively lower success rate in seen environments than the baseline without joint training. This indicates that the agent trained to take expert frames as instructions does not generalize well to human instructions. Using speaker translation annotations improves over the no joint training baseline by 10.6% and 21.8% in seen and unseen environments respectively. Furthermore, our proposed subgoal annotations bring an even larger relative improvement of 13.9% and 68.7% in seen and unseen environments which highlights the benefits of joint training with synthetic instructions in the form of subgoal actions.

Finally, we study if the recurrent baseline also benefits from joint training with synthetic data. Table 5 shows that the relative gains of joint training are 2.3 and 4.4 times higher for E.T. than for the LSTM-based agent in seen and unseen environments respectively. These numbers clearly show that E.T. benefits more from additional data and confirms the advantage of our model over LSTM-based agents.

Language encoder pretraining. Another application of synthetic instructions is to use them as an intermediate representation that decouples the visual appearance of an environment from the variations of human-annotated instructions. For this purpose, we pretrain the E.T. language encoder with the synthetic instructions. In particular, we pretrain a seq2seq model to map human instructions into synthetic instructions as described in Section 3.3, and study whether it is more beneficial to transfer explicitly the "translated" text or implicitly as representations encoded by the model weights. Our pretraining is done on the original train fold with no additionally generated trajectories. The seq2seq translation performance is very competitive, reaching 97.1% in terms of F1 score. To transfer explicitly the translated (synthetic) instructions, we first train an E.T. agent to follow synthetic instructions on the training fold and then evaluate the agent on following human instructions by translating these instructions into synthetic ones with our pretrained seq2seq model.

Objective	Transfer	Seen	Unseen
None	-	33.8	3.2
BERT	Text embedding	32.3	3.4
Seq2seq	Translated text	35.2	3.6
Seq2seq	Text encoder	**37.6**	**3.8**

Table 6: Comparison of models **with different language encoder pretraining** strategies. We pretrain a seq2seq model to map human instructions into synthetic instructions and transfer either its output text (third row) or its learned weights (fourth row). For completeness, we also compare with no pretraining (first row) and BERT pretraining (second row).

Table 6 compares these two pretraining strategies. We can see that both strategies outperform the no pretraining baseline (first row) significantly and that transferring the encoder works better than explicit translation. For completeness, we also report results with BERT pretraining [17] (second row). The BERT model is pretrained on generic text data (*e.g.* Wikipedia). We use the BERT base model whose weights are released by the authors. We extract its output contextualized word embeddings and use them as the input word embeddings to the language encoder. To our surprise, when compared with the no pretraining baseline, the BERT pretraining decreases the performance in seen environments by 4.4% and brings a marginal improvement of 6.2% relative in unseen environments. We conjecture that domain-specific language pretraining is important for the ALFRED benchmark. Overall, these experiments show another advantage of the proposed synthetic annotations and highlight the importance of intermediate language representations to better train instruction-following agents.

We finally combine the language encoder pretraining and the joint training objectives and present the results in Table 7. We observe that these two strategies are complementary to each other: the overall relative improvements of incorporating synthetic data over the baseline E.T. model are 37.8% and 228.1% in seen and unseen environments, respectively. We conclude that synthetic data is especially important for generalization to unseen environments. A complete breakdown of performance improvements can be found in Appendix A.2 of [49].

4.4. Comparison with state-of-the-art

We compare the E.T. agent with models with associated tech reports on the public leaderboard[2]. The results on validation and test folds are shown in Table 8. The complete table with *solved goal conditions* and *path-length-weighted scores* [2] is given in Appendix A.1 of [49] . The E.T. model trained without synthetic data pretraining and joint training sets a new state-of-the-art on seen environments (row 4). By leveraging synthetic instructions for pretraining, our method outperforms the previous methods [56, 58, 64] and sets a new state-of-the-art on all metrics (row 5). Given additional 45K trajectories for joint training, the E.T. model further improves the results (row 6).

[2] https://leaderboard.allenai.org/alfred, the results were submitted on February 22, 2021.

Pretraining	Joint training	Seen	Unseen
		33.8	3.2
✓		37.6	3.8
	✓	38.5	5.4
✓	✓	**46.6**	**7.3**

Table 7: Ablation study of **joint training and language encoder pretraining** with synthetic data. We present baseline results without leveraging synthetic data (first row), the independent performance of pretraining (second row) and joint training (third row), and their combined performance (fourth row).

Model	Validation		Test	
	Seen	Unseen	Seen	Unseen
Shridhar *et al.* [56]	3.70	0.00	3.98	0.39
Nguyen *et al.* [64]	N/A	N/A	12.39	4.45
Singh *et al.* [58]	19.15	3.78	22.05	5.30
E.T.	33.78	3.17	28.77	5.04
E.T. (pretr.)	37.63	3.76	33.46	5.56
E.T. (pretr. & joint tr.)	**46.59**	**7.32**	**38.42**	**8.57**
Human performance	-	-	-	91.00

Table 8: Comparison with the models submitted to the public leaderboard on validation and test folds. The highest value per fold is shown in **blue**. 'N/A' denotes that the scores are not reported on the leaderboard or in an associated publication. Our method sets a new state-of-the-art on all metrics.

5. Conclusion

We propose E.T., a transformer-based architecture for vision-and-language navigation tasks. E.T. observes the full episode history of vision, language, and action inputs and encodes it with a multimodal transformer. On the ALFRED benchmark, E.T. outperforms competitive recurrent baselines and achieves state-of-the-art performance on seen environments. We also propose to use synthetic instructions for pretraining and joint training with human-annotated instructions. Given the synthetic instructions, the performance is further improved in seen and especially, in unseen environments. In the future, we want to explore other forms of synthetic annotations and techniques to automatically construct them, for example with object detectors.

Acknowledgement: We thank Peter Anderson, Ellie Pavlick, and Dylan Ebert for helpful feedback on the draft.

References

[1] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. In *ACL*, 2020. 12

[2] Peter Anderson, Angel X. Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, and Amir Roshan Zamir. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018. 8, 12

[3] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, 2018. 1, 2

[4] Jacob Andreas and Dan Klein. Alignment-based compositional semantics for instruction following. In *EMNLP*, 2015. 2

[5] Yoav Artzi and Luke Zettlemoyer. Weakly supervised learning of semantic parsers for mapping instructions to actions. *TACL*, 2013. 3

[6] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, 2013. 3

[7] Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy, and Daniela Rus. Interpreting and executing recipes with a cooking robot. *Experimental Robotics*, 2013. 2

[8] Satchuthananthavale RK Branavan, Harr Chen, Luke S Zettlemoyer, and Regina Barzilay. Reinforcement learning for mapping instructions to actions. In *ACL*, 2009. 2

[9] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *NeurIPS*, 2020. 1, 2

[10] Guido Bugmann, Stanislao Lauria, Theocharis Kyriacou, Ewan Klein, Johan Bos, and Kenny Coventry. Using verbal instructions for route learning: Instruction analysis. *Proc. TIMR*, 2001. 2

[11] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 2

[12] David Chen and Raymond Mooney. Learning to interpret natural language navigation instructions from observations. In *AAAI*, 2011. 2

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

[14] Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, et al. Textworld: A learning environment for text-based games. In *Workshop on Computer Games*, 2018. 2

[15] Pim de Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. In *NeurIPS*, 2019. 6

[16] Matt Deitke, Winson Han, Alvaro Herrasti, Aniruddha Kembhavi, Eric Kolve, Roozbeh Mottaghi, Jordi Salvador, Dustin Schwenk, Eli VanderBilt, Matthew Wallingford, Luca Weihs, Mark Yatskar, and Ali Farhadi. Robothor: An open simulation-to-real embodied ai platform. In *CVPR*, 2020. 2

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 1, 2, 8

[18] David Ding, Felix Hill, Adam Santoro, and Matt Botvinick. Object-based attention for spatio-temporal reasoning: Outperforming neuro-symbolic models with flexible distributed architectures. *arXiv preprint arXiv:2012.08508*, 2020. 1

[19] 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 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2

[20] 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 *NeurIPS*, 2018. 2, 7

[21] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *ECCV*, 2020. 2

[22] M. Ghallab, A. Howe, C. Knoblock, D. Mcdermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins. PDDL: The Planning Domain Definition Language, 1998. 2, 4

[23] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In *CVPR*, 2019. 2

[24] Nakul Gopalan, Dilip Arumugam, Lawson Wong, and Stefanie Tellex. Sequence-to-sequence language grounding of non-markovian task specifications. In *RSS*, 2018. 2

[25] Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwińska, Sergio Gómez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, Adrià Puigdomènech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis. Hybrid computing using a neural network with dynamic external memory. *Nature*, 2016. 1

[26] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 5

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 4

[28] Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. A Recurrent Vision-and-Language BERT for Navigation. *CVPR*, 2021. 2

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

[30] Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *CVPR*, 2019. 2

[31] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv preprint arXiv:1712.05474*, 2017. 2, 5

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

[33] Alexander Ku, Peter Anderson, Roma Patel, Eugene Ie, and Jason Baldridge. Room-Across-Room: Multilingual Vision-and-Language Navigation with Dense Spatiotemporal Grounding. *EMNLP*, 2020. 2

[34] Juncheng Li, Xin Wang, Siliang Tang, Haizhou Shi, Fei Wu, Yueting Zhuang, and William Yang Wang. Unsupervised reinforcement learning of transferable meta-skills for embodied navigation. In *CVPR*, 2020. 2

[35] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 5

[36] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019. 2

[37] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *CoRL*, 2019. 2

[38] Corey Lynch and Pierre Sermanet. Grounding language in play. *RSS*, 2021. 2, 7

[39] Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. Self-monitoring navigation agent via auxiliary progress estimation. In *ICLR*, 2019. 1, 2, 6

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

[41] Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. Walk the talk: Connecting language, knowledge, and action in route instructions. In *AAAI*, 2006. 2

[42] Arjun Majumdar, Ayush Shrivastava, Stefan Lee, Peter Anderson, Devi Parikh, and Dhruv Batra. Improving vision-and-language navigation with image-text pairs from the web. In *ECCV*, 2020. 2

[43] Zohar Manna and Amir Pnueli. *The Temporal Logic of Reactive and Concurrent Systems*. Springer-Verlag, Berlin, Heidelberg, 1992. 2

[44] Harsh Mehta, Yoav Artzi, Jason Baldridge, Eugene Ie, and Piotr Mirowski. Retouchdown: Adding touchdown to streetlearn as a shareable resource for language grounding tasks in street view. *arXiv preprint arXiv:2001.03671*, 2020. 2

[45] Hongyuan Mei, Mohit Bansal, and Matthew Walter. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences. In *AAAI*, 2016. 2

[46] Dipendra Misra, John Langford, and Yoav Artzi. Mapping instructions and visual observations to actions with reinforcement learning. In *EMNLP*, 2017. 2

[47] Dipendra K Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. Tell me dave: Context-sensitive grounding of natural language to manipulation instructions. *The International Journal of Robotics Research*, 2016. 2

[48] Emilio Parisotto, Francis Song, Jack Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant Jayakumar, Max Jaderberg, Raphaël Lopez Kaufman, Aidan Clark, Seb Noury, Matthew Botvinick, Nicolas Heess, and Raia Hadsell. Stabilizing transformers for reinforcement learning. In *ICML*, 2020. 2

[49] Alexander Pashevich, Cordelia Schmid, and Chen Sun. Episodic Transformer for Vision-and-Language Navigation – supplementary material. *ICCV*, 2021. 7, 8

[50] Roma Patel, Ellie Pavlick, and Stefanie Tellex. Grounding language to non-markovian tasks with no supervision of task specifications. In *RSS*, 2020. 3

[51] Roma Patel, Roma Pavlick, and Stefanie Tellex. Learning to ground language to temporal logical form. In *NAACL*, 2019. 2

[52] Rohan Paul, Jacob Arkin, Derya Aksaray, Nicholas Roy, and Thomas M Howard. Efficient grounding of abstract spatial concepts for natural language interaction with robot platforms. *The International Journal of Robotics Research*, 2018. 2

[53] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In *CVPR*, 2018. 1, 2

[54] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 5

[55] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *ICCV*, 2019. 2

[56] Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A Benchmark for Interpreting Grounded Instructions for Everyday Tasks. In *CVPR*, 2020. 1, 2, 5, 6, 8, 12

[57] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *ICLR*, 2021. 2

[58] Kunal Pratap Singh, Suvaansh Bhambri, Byeonghwi Kim, Roozbeh Mottaghi, and Jonghyun Choi. MOCA: A Modular Object-Centric Approach for Interactive Instruction Following. *arXiv preprint arXiv:2012.03208*, 2020. 5, 6, 8, 12

[59] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *ICCV*, 2019. 1, 2

[60] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*, 2019. 1, 2

[61] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew Walter, Ashis Banerjee, Seth Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *AAAI*, 2011. 2

[62] Moritz Tenorth, Daniel Nyga, and Michael Beetz. Understanding and executing instructions for everyday manipulation tasks from the world wide web. In *ICRA*, 2010. 2

[63] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IROS*, 2012. 2

[64] Takayuki Okatani Van-Quang Nguyen. A hierarchical attention model for action learning from realistic environments and directives. In *ECCV EVAL Workshop*, 2020. 5, 8, 12

[65] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 2, 3, 4

[66] Oriol Vinyals, Lukasz Kaiser, Terry Koo, Slav Petrov, Ilya Sutskever, and Geoffrey Hinton. Grammar as a foreign language. In *NeurIPS*, 2015. 1, 2

[67] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 2

[68] Xin Wang, Qiuyuan Huang, Celikyilmaz Asli, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *CVPR*, 2019. 1, 2

[69] Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*, 2018. 3

[70] John M Zelle and Raymond J Mooney. Learning to parse database queries using inductive logic programming. In *AAAI*, 1996. 3

[71] Luke Zettlemoyer and Michael Collins. Online learning of relaxed ccg grammars for parsing to logical form. In *EMNLP*, 2007. 3

[72] Luke S Zettlemoyer and Michael Collins. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *arXiv preprint arXiv:1207.1420*, 2012. 3

[73] Fengda Zhu, Yi Zhu, Xiaojun Chang, and Xiaodan Liang. Vision-language navigation with self-supervised auxiliary reasoning tasks. In *CVPR*, 2020. 1

[74] Wang Zhu, Hexiang Hu, Jiacheng Chen, Zhiwei Deng, Vihan Jain, Eugene Ie, and Fei Sha. Babywalk: Going farther in vision-and-language navigation by taking baby steps. In *ACL*, 2020. 2