

ALFRED 

A Benchmark for Interpreting Grounded Instructions for Everyday Tasks

Mohit Shridhar¹
Winson Han³Jesse Thomason¹
Roozbeh Mottaghi^{1,3}Daniel Gordon¹
Luke Zettlemoyer¹Yonatan Bisk^{1,2,3}
Dieter Fox^{1,4}

AskForALFRED.com

Abstract

We present **ALFRED** (Action Learning From Realistic Environments and Directives), a benchmark for learning a mapping from natural language instructions and ego-centric vision to sequences of actions for household tasks. ALFRED includes long, compositional tasks with non-reversible state changes to shrink the gap between research benchmarks and real-world applications. ALFRED consists of expert demonstrations in interactive visual environments for 25k natural language directives. These directives contain both high-level goals like “Rinse off a mug and place it in the coffee maker.” and low-level language instructions like “Walk to the coffee maker on the right.” ALFRED tasks are more complex in terms of sequence length, action space, and language than existing vision-and-language task datasets. We show that a baseline model based on recent embodied vision-and-language tasks performs poorly on ALFRED, suggesting that there is significant room for developing innovative grounded visual language understanding models with this benchmark.

1. Introduction

A robot operating in human spaces must learn to connect natural language to the world. This *symbol grounding* [21] problem has largely focused on connecting language to static images. However, robots need to understand task-oriented language, for example “Rinse off a mug and place it in the coffee maker,” as illustrated in Figure 1.

Platforms for translating language to action have become increasingly popular, spawning new test-beds [3, 12, 14, 41]. These benchmarks include language-driven navigation and embodied question answering, which have seen dramatic improvements in modeling thanks to environments like Matterport 3D [3, 11], AI2-THOR [25], and AI Habitat [44]. However, these datasets ignore complexities arising from describing task-oriented behaviors with objects.

We introduce **ALFRED**, a new benchmark for con-



Figure 1: ALFRED consists of 25k language directives corresponding to expert demonstrations of household tasks. We highlight several frames corresponding to portions of the accompanying language instruction. ALFRED involves interactions with objects, keeping track of state changes, and references to previous instructions.

necting human language to *actions*, *behaviors*, and *objects* in interactive visual environments. Planner-based expert demonstrations are accompanied by both high- and low-level human language instructions in 120 indoor scenes in AI2-THOR 2.0 [25]. These demonstrations involve partial observability, long action horizons, underspecified natural language, and irreversible actions.

ALFRED includes 25,743 English language directives describing 8,055 expert demonstrations averaging 50 steps each, resulting in 428,322 image-action pairs. Motivated by work in robotics on segmentation-based grasping [36], agents in ALFRED interact with objects visually, specifying a pixelwise interaction mask of the target object. This inference is more realistic than simple object class prediction, where localization is treated as a solved problem. Existing beam-search [17, 47, 52] and backtracking solutions [24, 28] are infeasible due to the larger action and state spaces, long horizon, and inability to undo certain actions.

¹Paul G. Allen School of Computer Sci. & Eng., Univ. of Washington

²Language Technologies Institute @ Carnegie Mellon University

³Allen Institute for AI ⁴NVIDIA

	— Language —		— Virtual Environment —			— Inference —		
	# Human Annotations	Granularity	Visual Quality	Movable Objects	State Changes	Vis. Obs.	Navigation	Interaction
TACoS [42]	17k+	High&Low	Photos	✗	✗	–	–	–
R2R [3]; Touchdown [14]	21k+; 9.3k+	Low	Photos	✗	✗	Ego	Graph	✗
EQA [15]	✗	High	Low	✗	✗	Ego	Discrete	✗
Matterport EQA [54]	✗	High	Photos	✗	✗	Ego	Discrete	✗
IQA [20]	✗	High	High	✗	✓	Ego	Discrete	Discrete
VirtualHome [41]	2.7k+	High&Low	High	✓	✓	3 rd Person	✗	Discrete
VSP [57]	✗	High	High	✓	✓	Ego	✗	Discrete
ALFRED 🤖	25k+	High&Low	High	✓	✓	Ego	Discrete	Discrete + Mask

Table 1: **Dataset comparison.** ALFRED is the first interactive visual dataset to include high-level goal and low-level natural language instructions for object and environment interactions. TACoS [42] provides detailed high- and low-level text descriptions of cooking videos, but does not facilitate task execution. For navigation, ALFRED enables discretized, grid-based movement, while other datasets use topological graph navigation or avoid navigation altogether. ALFRED requires an agent to generate spatially located interaction masks for action commands. By contrast, other datasets only require choosing from a discrete set of available interactions and object classes or offer no interactive capability.

To establish baseline performance levels, we evaluate a sequence-to-sequence model akin to existing vision-and-language navigation tasks [27]. This model is not effective on the complex tasks in ALFRED, achieving less than 5% success rates. For analysis, we also evaluate individual sub-goals. While performance is better for isolated sub-goals, the model lacks the reasoning capacity for long-horizon and compositional task planning.

In summary, ALFRED facilitates learning models that translate from language to sequences of actions and interactions in a visually and physically realistic simulation environment. This benchmark captures many challenges present in real-world settings for translating human language to robot actions for accomplishing household tasks. Models that can overcome these challenges will begin to close the gap towards real-world, language-driven robotics.

2. Related Work

Table 1 summarizes the benefits of ALFRED relative to other visual action datasets with language annotations.

Vision & Language Navigation. In vision-and-language navigation tasks, either natural or templated language describes a route to a goal location through egocentric visual observations [3, 12, 13, 14, 30]. Since the proposal of R2R [3], researchers have dramatically improved the navigation performance of models [17, 24, 28, 52, 53] with techniques like progress monitoring [27], as well as introduced task variants with additional, on-route instructions [37, 38, 50]. Much of this research is limited to static environments. By contrast, ALFRED tasks include navigation, object interactions, and state changes.

Vision & Language Task Completion. There are sev-

eral existing benchmarks based on simple block worlds and fully observable scenes [9, 33]. ALFRED provides more difficult tasks in richer, visually complex scenes, and uses partially observable environments. The CHAI benchmark [32] evaluates agents performing household instructions, but uses a generic “interact” action. ALFRED has seven manipulation actions, such as pick up, turn on, and open, state changes like clean versus dirty, and variation in language and visual complexity.

Previous work in the original AI2-THOR environment investigated the task of visual semantic planning [19, 57]. Artificial language came from templates, and environment interaction was handled with discrete class predictions, for example selecting *apple* as the target object from predefined options. ALFRED features human language instructions, and object selections are carried out with class-agnostic, pixelwise interaction masks. In VirtualHome [41], programs are generated from video demonstration and natural language instructions, but inference does not involve egocentric visual and action feedback or partial observability.

There is an extensive literature on language-based instruction following in the natural language processing community. There, research has focused on mapping instructions to actions [5, 13, 31, 35, 48], but these works do not involve visual, interactive environments.

Embodied Question Answering. Existing datasets for visual question answering in embodied environments use templated language or static scenes [15, 20, 54, 56]. In ALFRED, rather than answering a question, the agent must complete a task specified using natural language, which requires both navigation and interaction with objects.

Instruction Alignment. Language annotations of videos enable discovering visual correspondences between words









	Pick & Place	Stack & Place	Pick Two & Place	Clean & Place	Heat & Place	Cool & Place	Examine in Light
item(s)	Book	Fork (in) Cup	Spray Bottle	Dish Sponge	Potato Slice	Egg	Credit Card
receptacle	Desk	Counter Top	Toilet Tank	Cart	Counter Top	Side Table	Desk Lamp
scene #	Bedroom 14	Kitchen 10	Bathroom 2	Bathroom 1	Kitchen 8	Kitchen 21	Bedroom 24
expert demonstration							
				Annotation # 1 Goals Put a clean sponge on a metal rack. Instructions Go to the left and face the faucet side of the bath tub. Pick up left most green sponge from the bath tub. Turn around and go to the sink. Put the sponge in the sink. Turn on then turn off the water. Take the sponge from the sink. Go to the metal bar rack to the left. Put the sponge on the top rack to the left of the lotion bottle.	Annotation # 2 Goals Place a clean sponge on the drying rack Instructions Turn around and walk over to the bathtub on the left. Grab the sponge out of the bathtub. Turn around and walk to the sink ahead. Rinse the sponge out in the sink. Move to the left a bit and face the drying rack in the corner of the room. Place the sponge on the drying rack.	Annotation # 3 Goals Put a rinsed out sponge on the drying rack Instructions Walk forwards a bit and turn left to face the bathtub. Grab a sponge out of the bathtub. Turn around and walk forwards to the sink. Rinse the sponge out in the sink and pick it up again. Turn left to walk a bit, then face the drying rack. Put the sponge on the drying rack.	

Figure 2: **ALFRED annotations.** We introduce 7 different task types parameterized by 84 object classes in 120 scenes. An example of each task type is given above. For the **Clean & Place** demonstration, we also show the three crowdsourced language directives. Please see the supplemental material for example demonstrations and language for each task.

and concepts [1, 45, 42, 55, 58]. ALFRED requires performing tasks in an interactive setting as opposed to learning from recorded videos.

Robotics Instruction Following. Instruction following is a long-standing topic of interest in robotics [7, 10, 29, 34, 39, 40, 46, 51]. Lines of research consider different tasks such as cooking [10], table clearing [39], and mobile manipulation [29]. In general, they are limited to a few scenes [34], consider a small number of objects [29], or use the same environment for training and testing [7]. In contrast, ALFRED includes 120 scenes, many object classes with diverse appearances, and a test set of unseen environments.

3. The ALFRED Dataset

The ALFRED dataset comprises 25,743 language directives corresponding to 8,055 expert demonstration episodes. Each directive includes a high-level goal and a set of step-by-step instructions. Each expert demonstration can be deterministically replayed in the AI2-THOR 2.0 simulator.

3.1. Expert Demonstrations

Expert demonstrations are composed of an agent’s egocentric visual observations of the environment and what action is taken at each timestep as well as ground-truth interaction masks. These demonstrations are generated by a planner [23] using metadata not available to the agent at inference time. Navigation actions move the agent or change its camera orientation, while manipulation actions include picking and placing objects, opening and closing cabinets and drawers, and turning appliances on and off. Interactions can involve multiple objects, such as using a knife to slice an apple, cleaning a mug in the sink, and heating a potato in the microwave. Manipulation actions are accompanied by a ground truth segmentation of the target object.

Figure 2 gives examples of the high-level agent tasks in ALFRED, like putting a cleaned object at a destination. These tasks are parameterized by the object of focus, the destination receptacle (e.g., *table top*), the scene in which to carry out the task, and in the case of **Stack & Place**, a base object (e.g., *plate*). ALFRED contains expert demonstrations of these seven tasks executed using combinations of 58 unique object classes and 26 receptacle object classes across 120 different indoor scenes. For object classes like *potato slice*, the agent must first pick up a *knife* and find a *potato* to create slices. All object classes contain multiple visual variations with different shapes, textures, and colors. For example, there are 30 unique variants of the *apple* class. Indoor scenes include different room types: 30 each of kitchens, bathrooms, bedrooms, and living rooms.

For 2,685 combinations of task parameters, we generate three expert demonstrations per parameter set, for a total of 8,055 unique demonstrations with an average of 50 action steps. The distributions of actions steps in ALFRED demonstrations versus related datasets is given in Figure 3. As an example, for task parameters {task: **Heat & Place**, object: *potato*, destination: *counter top*, scene: KITCHEN-8}, we generate three different expert demonstrations by starting the agent and objects in randomly chosen locations. Object start positions have some commonsense, class-specific constraints, for example a *fork* can start inside a *drawer*, but an *apple* cannot.

Contrasting navigation-only datasets where expert demonstrations can come from an A^* planner, our state space includes object positions and state changes. Thus, to generate expert demonstrations we encode the agent and object states, as well as high-level environment dynamics, into Planning Domain Definition Language (PDDL) rules [18]. We then define task-specific PDDL goal conditions, for example that a heated *potato* is resting on a *table top*. Note

	Train	Validation		Test	
		<i>Seen</i>	<i>Unseen</i>	<i>Seen</i>	<i>Unseen</i>
# Annotations	21,023	820	821	1,533	1,529
# Scenes	108	88	4	107	8

Table 2: **ALFRED Data Splits.** All expert demonstrations and associated language directives in the validation and test folds are distinct from those in the train fold. The validation and test sets are split into *seen* and *unseen* folds. Scenes in the *seen* folds of validation and test data are subsets of those in the train fold. Scenes in the *unseen* validation and test folds are distinct from the train folds and from each other.

that the planner encodes the environment as *fully observable* and has perfect knowledge about world dynamics. For training and testing agent models, however, the environment is *partially observable*: it is only viewed through the agent’s egocentric vision as actions are carried out.

We split these expert demonstrations into training, validation, and test folds (Table 2). Following work in vision-and-language navigation [3], we further split the validation and test into two conditions: *seen* and *unseen* environments. This split facilitates examining how well models generalize to entirely new spaces with novel object class variations.

3.2. Language Directives

For every expert demonstration, we collect open vocabulary, free-form language directives from at least three different annotators using Amazon Mechanical Turk (AMT), resulting in 25k total language directives. Language directives include a high-level *goal* together with low-level *instructions*, as shown in Figures 1 and 2. The distribution of language annotation token lengths in ALFRED versus related datasets is given in Figure 3.

AMT workers are told to write instructions to tell a “smart robot” how to accomplish what is shown in a video. We create a video of each expert demonstration and segment it such that each segment corresponds to an instruction. We consult the PDDL plan for the expert demonstration to identify task sub-goals, for example the many low-level steps to navigate to a knife, or the several steps to heat a potato slice in the microwave once standing in front of it. We visually highlight action sequences related to sub-goals via colored timeline bars below the video. In each HIT (Human Intelligence Task), a worker watches the video, then writes low-level, step-by-step *instructions* for each highlighted sub-goal segment. The worker also writes a high-level *goal* that summarizes what the robot should accomplish during the expert demonstration.

These directives are validated through a second HIT by at least two annotators, with a possible third tie-breaker. For validation, we show a worker all three language directive

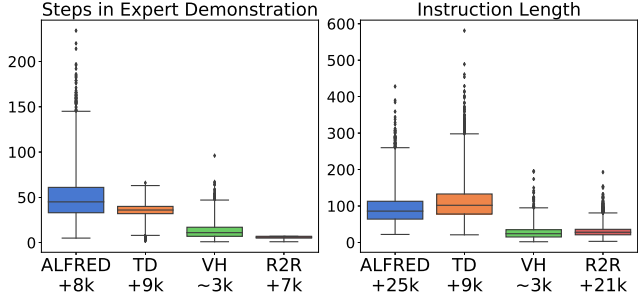


Figure 3: **Comparison to Existing Datasets.** Expert demonstration steps and instruction tokens of ALFRED compared to other datasets with human language for action sequences: Touchdown (TD) [14], VirtualHome (VH) [41], and Room-to-Room (R2R) [3]. The total number of demonstrations or annotations is given with the dataset label.

annotations without the video. The worker selects whether the three directives describe the same actions, and if not, which is most different. If a directive is chosen as most different by a majority of validation workers, it is removed and the demonstration is subsequently re-annotated by another worker. Qualitatively, these rejected annotations contain incorrect object referents (*e.g.*, “egg” instead of “potato”) or directions (*e.g.*, “go left towards...” instead of “right”).

4. Baseline Models

An agent trained for ALFRED tasks needs to jointly reason over vision and language input and produce a sequence of low-level actions to interact with the environment.

4.1. Sequence-to-Sequence Models

We model the interactive agent with a CNN-LSTM sequence-to-sequence (SEQ2SEQ) architecture. A CNN encodes the visual input, a bidirectional-LSTM generates a representation of the language input, and a decoder LSTM infers a sequence of low-level actions while attending over the encoded language. See Figure 4 for an overview and the supplementary material for implementation details.

Supervision. We train all models using imitation learning on expert trajectories. This ensures the language directives match the visual inputs. At each timestep, the model is trained to produce the expert action and associated interaction mask for manipulation actions.

We note that a DAGger-style [43] student-forcing paradigm in ALFRED is non-trivial, even disregarding language alignment. Obtaining expert demonstration actions on the fly in navigation-only datasets like R2R [3] only requires rerunning A^* . In ALFRED, on the fly demonstrations requires re-planning. In some cases re-planning is not

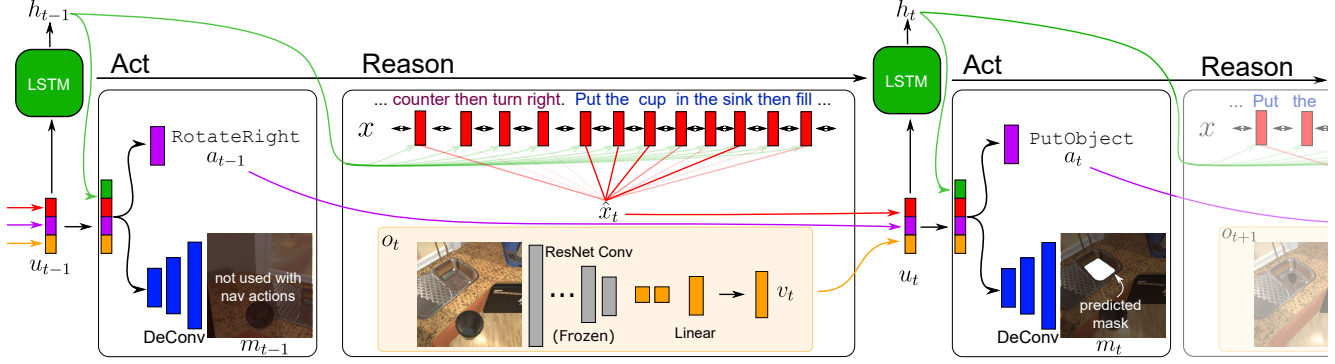


Figure 4: **Model overview.** At each step, our model reweights the instruction based on the history (\hat{x}_t), and combines the current observation features (v_t) and the previously executed action (a_{t-1}). These are passed as input to an LSTM cell to produce the current hidden state. Finally, the new hidden state (h_t) is combined with the previous features to predict both the next action (a_t) and a pixelwise interaction mask over the observed image to indicate an object.

possible: if during a task of $\{\text{Clean \& Place, apple, refrigerator, KITCHEN-3}\}$ a student-forcing model slices the only *apple* in the scene, the action cannot be recovered from and the task cannot be completed.

Visual encoding. Each visual observation o_t is encoded with a frozen ResNet-18 [22] CNN, where we take the output of the final convolution layer to preserve spatial information necessary for grounding specific objects in the visual frame. We embed this output using two more 1×1 convolution layers and a fully-connected layer. During training, a set of T observations from the expert demonstration is encoded as $\bar{V} = \langle v_1, v_2, \dots, v_T \rangle$, where v_t is the visual feature vector at time-step t .

Language encoding. Given a natural language goal $\bar{G} = \langle g_1, g_2, \dots, g_{L_g} \rangle$ of L_g words, and step-by-step instructions $\bar{S} = \langle s_1, s_2, \dots, s_{L_s} \rangle$ of L_s words, we append them into a single input sequence $\bar{X} = \langle g_1, g_2, \dots, g_{L_g}, \langle \text{SEP} \rangle, s_1, s_2, \dots, s_{L_s} \rangle$ with the $\langle \text{SEP} \rangle$ token indicating the separation between the high-level goal and low-level instructions. This sequence is fed into a bi-directional LSTM encoder to produce an encoding $x = \{x_1, x_2, \dots, x_{L_g+L_s}\}$ for each word in \bar{X} .

Attention over language. The agent’s action at each timestep is based on an attention mechanism weighting tokens in the instruction. We perform soft-attention on the language features x to compute the attention distribution α_t conditioned on the hidden state of the decoder h_{t-1} from the last timestep:

$$\begin{aligned} z_t &= (W_x h_{t-1})^\top x, \\ \alpha_t &= \text{Softmax}(z_t), \\ \hat{x}_t &= \alpha_t^\top x \end{aligned} \quad (1)$$

where W_x are learnable parameters of a fully-connected layer, z_t is a vector of scalar values that represent the attention mass for each word in x , and \hat{x}_t is the weighted sum of x over the attention distribution α_t induced from z_t .

Action decoding. At each timestep t , upon receiving a new observation image o_t , the LSTM decoder takes in the visual feature v_t , language feature \hat{x}_t , and the previous action a_{t-1} , and outputs a new hidden state h_t :

$$\begin{aligned} u_t &= [v_t; \hat{x}_t; a_{t-1}], \\ h_t &= \text{LSTM}(u_t, h_{t-1}) \end{aligned} \quad (2)$$

where $[\cdot]$ denotes concatenation. The hidden state h_t is used to obtain the attention weighted language feature \hat{x}_{t+1} .

Action and mask prediction. The agent interacts with the environment by choosing an action and producing a pixelwise binary mask indicating a specific object in the frame. Although AI2-THOR supports continuous control for agent navigation and object manipulation, we discretize the action space. The agent chooses from among 13 actions. There are 5 navigation actions: *MoveAhead*, *RotateRight*, *RotateLeft*, *LookUp*, and *LookDown* together with 7 interaction actions: *Pickup*, *Put*, *Open*, *Close*, *ToggleOn*, *ToggleOff*, and *Slice*. Interaction actions require a pixelwise mask to denote the object of interest.¹ Finally, the agent predicts a *Stop* action to end the episode. We concatenate the hidden state h_t with the input features u_t and train two separate networks to predict the next action

¹The final object chosen by the interaction API is based on the Intersection-over-Union (IoU) score between the predicted mask and the ground-truth object mask from the simulator.

a_t and interaction mask m_t :

$$\begin{aligned} a_t &= \operatorname{argmax} (W_a [h_t; u_t]), \\ m_t &= \sigma (\mathbf{deconv} [h_t; u_t]) \end{aligned} \quad (3)$$

where W_a are learnable parameters of a fully connected layer, \mathbf{deconv} is a three-layer deconvolution network, and σ is a sigmoid activation function. Action selection is trained using softmax cross entropy with the expert action. The interaction masks are learned end-to-end in a supervised manner based on ground-truth object segmentations using binary cross-entropy loss. The mask loss is rebalanced to account for sparsity in these dense masks in which target objects can take up a small portion of the visual frame.

4.2. Progress Monitors

ALFRED tasks require reasoning over long sequences of images and instruction words. We propose two auxiliary losses (Eq. 4 & 5) that use additional temporal information to reduce this burden and form a sequence-to-sequence model with progress monitoring (SEQ2SEQ+PM).

Ma *et al.* [27] showed that agents benefit from maintaining an internal estimate of their progress towards the goal for navigation tasks. Akin to learning a value function in reinforcement learning, progress monitoring helps to learn the utility of each state in the process of achieving the overall task. Intuitively, this allows our agent to better distinguish between visually similar states such as just before putting an object in the microwave versus just after taking the object out. We introduce a simple module that predicts progress, $p_t \in [0, 1]$, conditioned on the decoder hidden state h_t and the concatenated input u_t :

$$p_t = \sigma (W_p [h_t; u_t]). \quad (4)$$

The supervision for p_t is based on normalized time-step values t/T , where t is the current time-step, and T is the total length of the expert demonstration (trained via L2 loss).

We also train the agent to predict the number of sub-goals completed so far, c_t . These sub-goals represent segments in the demonstration corresponding to sequences of actions like navigation, pickup, and heating as identified in the PDDL plan, discussed in Section 3.2. Each segment has a corresponding language instruction, but the alignment must be learned. This sub-goal prediction encourages the agent to coarsely track its progress through the language directive. This prediction is also conditioned on the decoder hidden state h_t and the concatenated input u_t :

$$c_t = \sigma (W_c [h_t; u_t]). \quad (5)$$

We train c_t in a supervised fashion by using the normalized number of sub-goals accomplished in the expert trajectory at each timestep, c_t/C , as the ground-truth label for a task with C sub-goals. We again train with an L2 loss.

5. Experiments

We evaluate the baseline models in the AI2-THOR simulator. When evaluating on test folds, we run models with the lowest validation loss. Episodes that exceed 1000 steps or cause more than 10 failed actions are terminated. Failed actions arise from bumping into walls or predicting action interaction masks for incompatible objects, such as attempting to `Pickup` a *counter top*. These limitations encourage efficiency and reliability. We assess the overall and partial success of models’ task executions across episodes.

5.1. Evaluation Metrics

ALFRED allows us to evaluate both full task and task goal-condition completion. In navigation-only tasks, one can only measure how far the agent is from the goal. In ALFRED, we can also evaluate whether task goal-conditions have been completed, for example that a *potato* has been sliced. For all of our experiments, we report both Task Success and Goal-Condition Success. Each Goal-Condition relies on multiple instructions, for example navigating to an object and then slicing it.

Task Success. Each expert demonstration is parameterized by a task to be performed, as illustrated in Figure 2. Task Success is defined as 1 if the object positions and state changes correspond correctly to the task goal-conditions at the end of the action sequence, and 0 otherwise. Consider the task: “Put a hot potato slice on the counter”. The agent succeeds if, at the end of the episode, any *potato slice* object has changed to the *heated* state and is resting on any *counter top* surface.

Goal-Condition Success. The goal-condition success of a model is the ratio of goal-conditions completed at the end of an episode to those necessary to have finished a task. For example, in the previous **Heat & Place** example, there are four goal-conditions. First, a *potato* must be sliced. Second, a *potato slice* should become *heated*. Third, a *potato slice* should come to rest on a *counter top*. Fourth, the same *potato slice* that is *heated* should be on the *counter top*. If the agent slices a *potato*, then moves a slice to the counter top without heating it, then the goal-condition success score is $2/4 = 50\%$. On average, tasks in ALFRED have 2.55 goal conditions. The final score is calculated as the average goal-condition success of each episode. Task success is 1 only if goal-condition success is 1.

Path Weighted Metrics. We include a Path Weighted version of both metrics that considers the length of the expert demonstration [2]. Expert demonstrations found via a PDDL solver on global information are not guaranteed to be optimal. However, they avoid exploration, use shortest path

Model	Validation				Test			
	Seen		Unseen		Seen		Unseen	
	Task	Goal-Cond	Task	Goal-Cond	Task	Goal-Cond	Task	Goal-Cond
NO LANGUAGE	0.0 (0.0)	5.9 (3.4)	0.0 (0.0)	6.5 (4.7)	0.2 (0.0)	5.0 (3.2)	0.2 (0.0)	6.6 (4.0)
NO VISION	0.0 (0.0)	5.7 (4.7)	0.0 (0.0)	6.8 (6.0)	0.0 (0.0)	3.9 (3.2)	0.2 (0.1)	6.6 (4.6)
GOAL-ONLY	0.1 (0.0)	6.5 (4.3)	0.0 (0.0)	6.8 (5.0)	0.1 (0.1)	5.0 (3.7)	0.2 (0.0)	6.9 (4.4)
INSTRUCTIONS-ONLY	2.3 (1.1)	9.4 (6.1)	0.0 (0.0)	7.0 (4.9)	2.7 (1.4)	8.2 (5.5)	0.5 (0.2)	7.2 (4.6)
SEQ2SEQ	2.4 (1.1)	9.4 (5.7)	0.1 (0.0)	6.8 (4.7)	2.1 (1.0)	7.4 (4.7)	0.5 (0.2)	7.1 (4.5)
+ PM PROGRESS-ONLY	2.1 (1.1)	8.7 (5.6)	0.0 (0.0)	6.9 (5.0)	3.0 (1.7)	8.0 (5.5)	0.3 (0.1)	7.3 (4.5)
+ PM SUBGOAL-ONLY	2.1 (1.2)	9.6 (5.5)	0.0 (0.0)	6.6 (4.6)	3.8 (1.7)	8.9 (5.6)	0.5 (0.2)	7.1 (4.5)
+ PM Both	3.7 (2.1)	10.0 (7.0)	0.0 (0.0)	6.9 (5.1)	4.0 (2.0)	9.4 (6.3)	0.4 (0.1)	7.0 (4.3)
HUMAN	-	-	-	-	-	-	91.0 (85.8)	94.5 (87.6)

Table 3: **Task and Goal-Condition Success.** For each metric, the corresponding path weighted metrics are given in parentheses. The highest values per fold and metric are shown in **blue**. All values are percentages.

navigation, and are generally efficient. The path weighted score p_s for metric s is given as

$$p_s = s \times \frac{L^*}{\max(L^*, \hat{L})} \quad (6)$$

where \hat{L} is the number of actions the model took in the episode, and L^* is the number of actions in the expert demonstration. Intuitively, a model receives half-credit for taking twice as long as the expert to accomplish a task.

5.2. Sub-Goal Evaluation

Completing the entire sequence of actions required to finish a task is challenging. In addition to assessing full task success, we study the ability of a model to accomplish the next sub-goal conditioned on the preceding expert sequence. The agent is tested by first forcing it to follow the expert demonstration to maintain a history of states leading up to the sub-goal, then requiring it to complete the sub-goal conditioned on the entire language directive and current visual observation. For the task ‘‘Put a hot potato slice on the counter’’ for example, we can evaluate the sub-goal of navigating to the potato after using the expert demonstration to navigate to and pick up a *knife*. The tasks in ALFRED contain on average 7.5 such sub-goals (results in Table 4).

6. Analysis

Results from our experiments are presented in Table 3. We find that the initial model, without spatial or semantic maps, object segmentations, or explicit object-state tracking, performs poorly on ALFRED’s long-horizon tasks with high-dimensional state-spaces. The SEQ2SEQ model achieves $\sim 8\%$ goal-condition success rate, showing that the agent does learn to partially complete some tasks. This headroom (as compared with humans) motivates further research into models that can perform the complex vision-and-language planning introduced by ALFRED. The performance starkly contrasts other vision-and-

language datasets focused on navigation, where sequence-to-sequence with progress monitoring performs well [27].

6.1. Random Agent

A random agent is commonly employed as a baseline in vision-and-language tasks. In ALFRED, an agent that chooses a uniform random action and generates a uniform random interaction mask at each timestep achieves 0% on all folds, even without an API failure limit.

6.2. Unimodal Ablations

Previous work established that learned agents without visual inputs, language inputs, or both performed better than random agents and were competitive with initial baselines for several navigation and question answering tasks [49]. These performance gaps were due to structural biases in the datasets or issues with model capacity. We evaluate these ablation baselines (NO LANGUAGE and NO VISION) to study vision and language bias in ALFRED.

The unimodal ablation performances in Table 3 indicate that both vision and language modalities are necessary to accomplish the tasks in ALFRED. The NO LANGUAGE model finishes some goal-conditions by interacting with familiar objects seen during training. The NO VISION model similarly finishes some goal-conditions by following low-level language instructions for navigation and memorizing interaction masks for common objects like *microwaves* that are centered in the visual frame.

6.3. Model Ablations

We additionally ablate the amount of language supervision available to the model, as language directives are given as both a high-level goal and step-by-step instructions. Providing only high-level, underspecified goal language (GOAL-ONLY) is insufficient to complete the tasks, but is enough to complete some goal-conditions. Using just low-level, step-by-step instructions (INSTRUCTIONS-ONLY

performs similarly to using both high- and low-levels. Thus, this simple model does not seem to exploit the goal instruction to plan out sub-goals for step-by-step execution.

The two progress monitoring signals are marginally helpful, increasing the success rate from $\sim 1\%$ to $\sim 2\%$. Progress monitoring leads to more efficient task completion, as indicated by the consistently higher path weighted scores. They may help avoid action repetition and with the prediction of the `Stop` action.

The agent takes more steps than the expert in all cases, as indicated by the lower path weighted scores. Sometimes, this is caused by failing to keep track of state-changes, for example heating up an *egg* in the *microwave* multiple times. Further, the models also do not generalize well to *unseen* scenes, due to the overall visual complexity in ALFRED arising from new scenes and novel object class instances.

6.4. Human evaluation

We obtained a human evaluation of 100 randomly sampled directives from the *unseen* test fold. The experiment involved 5 participants who completed 20 tasks each using a keyboard-and-mouse interface. Before the experiment, the participants were allowed to familiarize themselves with AI2-THOR. The action-space and task restrictions were identical to that of the baseline models. Overall, the participants obtained a high success rate of 91%, while taking slightly longer than the expert with 86% path-length weighted success rate. This indicates that the directives in ALFRED are well-aligned with the demonstrations.

6.5. Sub-Goal Performance


We also examine performance of the SEQ2SEQ model on individual sub-goals in ALFRED. For this experiment, we use the expert trajectory to move the agent through the episode up to the sub-task. Then, the agent begins inference based on the language directive and current visual frame.

Table 4 presents path-length weighted success scores for 8 sub-goals. **Goto** and **Pickup** sub-tasks with the SEQ2SEQ+PM model achieve $\sim 51\%$ and $\sim 32\%$, respectively, even in *seen* environments. Visual semantic navigation is considerably harder in *unseen* environments. Similarly, interaction masks for **Pickup** actions in *unseen* environments are worse due to unfamiliar scenes and object instances. Simple sub-goals like **Cool**, and **Heat** are achieved at a high success rate of $\sim 90\%$ because these tasks are mostly object-agnostic. For example, the agent becomes familiar with using microwaves to heat things regardless of the object in-hand, because microwaves have little visual diversity across kitchens. Overall, the sub-goal evaluations indicate that models that exploit modularity and hierarchy, or make use of pretrained object segmentation models, may make headway on full task sequences.

		Sub-Goal Ablations - Validation								
Model		<i>Goto</i>	<i>Pickup</i>	<i>Put</i>	<i>Cool</i>	<i>Heat</i>	<i>Clean</i>	<i>Slice</i>	<i>Toggle</i>	Avg.
<i>Seen</i>	No Lang	28	22	71	89	87	64	19	90	59
	S2S	49	32	80	87	85	82	23	97	67
	S2S + PM	51	32	81	88	85	81	25	100	68
<i>Unseen</i>	No Lang	17	9	31	75	86	13	8	4	30
	S2S	21	20	51	94	88	21	14	54	45
	S2S + PM	22	21	46	92	89	57	12	32	46

Table 4: **Evaluations by path weighted sub-goal success.** All values are percentages. The highest values per fold and task are shown in **blue**. We note that the NO VISION model achieves less than 2% on all sub-goals. See supplemental material for more.

7. Conclusions

We introduced ALFRED , a benchmark for learning to map natural language instructions and egocentric vision to sequences of actions. ALFRED moves us closer to a community goal of language-driven robots capable of navigation and interaction. The environment dynamics and interaction mask predictions required in ALFRED narrow the gap between what is required of agents in simulation and robots operating in the real world [36].

We use ALFRED to evaluate a sequence-to-sequence model with progress monitoring, shown to be effective in other vision-and-language navigation tasks [27]. While this model is relatively competent at accomplishing some sub-goals (e.g. operating microwaves is similar across **Heat & Place** tasks), the overall task success rates are poor. The long horizon of ALFRED tasks poses a significant challenge with sub-problems including visual semantic navigation, object detection, referring expression grounding, and action grounding. These challenges may be approachable by models that exploit hierarchy [8, 26], modularity [4, 16], and structured reasoning and planning [6]. We are encouraged by the possibilities and challenges that the ALFRED benchmark introduces to the community.

Acknowledgements

Thanks to our UW colleagues for helpful feedback, to Eli VanderBilt and Eric Kolve for their help with AI2-THOR and leaderboard setup, and Victor Zhong for early modeling design. And finally, thanks to Ranjay Krishna for sharing the Mechanical Turk annotation interface. This research was supported in part by the ARO (ARO-W911NF-16-1-0121), the NSF (IIS1252835, IIS-1562364, NSF-NRI-1637479), and the Allen Institute for Artificial Intelligence.

References

- [1] Jean-Baptiste Alayrac, Piotr Bojanowski, Nishant Agrawal, Ivan Laptev, Josef Sivic, and Simon Lacoste-Julien. Unsupervised learning from narrated instruction videos. In *CVPR*, 2016.
- [2] 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.
- [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.
- [4] Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *CVPR*, June 2016.
- [5] Yoav Artzi and Luke Zettlemoyer. Weakly supervised learning of semantic parsers for mapping instructions to actions. *TACL*, 2013.
- [6] Masataro Asai and Alex Fukunaga. Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. In *AAAI*, 2018.
- [7] Michael Beetz, Ulrich Klank, Ingo Kresse, Alexis Maldonado, Lorenz Mösenlechner, Dejan Pangercic, Thomas Rühr, and Moritz Tenorth. Robotic roommates making pancakes. In *IEEE-RAS*, 2011.
- [8] Yonatan Bisk, Daniel Marcu, and William Wong. Towards a dataset for human computer communication via grounded language acquisition. In *AAAI Workshop on Symbiotic Cognitive Systems*, 2016.
- [9] Yonatan Bisk, Deniz Yuret, and Daniel Marcu. Natural language communication with robots. In *NAACL*, 2016.
- [10] Mario Bollini, Stefanie Tellex, Tyler Thompson, Nicholas Roy, and Daniela Rus. Interpreting and executing recipes with a cooking robot. In *ISER*, 2012.
- [11] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from RGB-D data in indoor environments. *3DV*, 2017.
- [12] Devendra Singh Chaplot, Kanthashree Mysore Sathyendra, Rama Kumar Pasumarthi, Dheeraj Rajagopal, and Ruslan Salakhutdinov. Gated-attention architectures for task-oriented language grounding. In *AAAI*, 2017.
- [13] David L Chen and Raymond J Mooney. Learning to interpret natural language navigation instructions from observations. In *AAAI*, 2011.
- [14] Howard Chen, Alane Suhr, Dipendra Misra, Noah Snaveley, and Yoav Artzi. Touchdown: Natural language navigation and spatial reasoning in visual street environments. In *CVPR*, 2019.
- [15] Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Embodied Question Answering. In *CVPR*, 2018.
- [16] Abhishek Das, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. Neural Modular Control for Embodied Question Answering. In *CoRL*, 2018.
- [17] 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.
- [18] Malik Ghallab, Adele Howe, Craig Knoblock, Drew McDermott, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. Pddl the planning domain definition language. 1998.
- [19] Daniel Gordon, Dieter Fox, and Ali Farhadi. What should i do now? marrying reinforcement learning and symbolic planning. *arXiv preprint arXiv:1901.01492*, 2018.
- [20] Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. Iqa: Visual question answering in interactive environments. In *CVPR*, 2017.
- [21] Stevan Harnad. The symbol grounding problem. *Physica D*, 42:335–346, 1990.
- [22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.
- [23] Jörg Hoffmann and Bernhard Nebel. The ff planning system: Fast plan generation through heuristic search. *JAIR*, 2001.
- [24] Liyiming Ke, Xiujuan 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.
- [25] 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.
- [26] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. In *NeurIPS*, pages 3675–3683, 2016.
- [27] 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.
- [28] Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. The regretful agent: Heuristic-aided navigation through progress estimation. In *CVPR*, 2019.
- [29] James MacGlashan, Monica Babes-Vroman, Marie desJardins, Michael L. Littman, Smaranda Muresan, Shawn Squire, Stefanie Tellex, Dilip Arumugam, and Lei Yang. Grounding english commands to reward functions. In *RSS*, 2015.
- [30] Matt MacMahon, Brian Stankiewicz, and Benjamin Kuipers. Walk the talk: Connecting language, knowledge, and action in route instructions. In *AAAI*, 2006.
- [31] Jonathan Malmaud, Earl Wagner, Nancy Chang, and Kevin Murphy. Cooking with semantics. In *ACL Workshop on Semantic Parsing*, 2014.
- [32] Dipendra Misra, Andrew Bennett, Valts Blukis, Eyvind Niklasson, Max Shatkhin, and Yoav Artzi. Mapping instructions to actions in 3D environments with visual goal prediction. In *EMNLP*, 2018.

- [33] Dipendra Misra, John Langford, and Yoav Artzi. Mapping instructions and visual observations to actions with reinforcement learning. In *EMNLP*, 2017.
- [34] Dipendra Misra, Jaeyong Sung, Kevin Lee, and Ashutosh Saxena. Tell me daved: Context-sensitive grounding of natural language to manipulation instructions. In *RSS*, 2014.
- [35] Dipendra Misra, Kejia Tao, Percy Liang, and Ashutosh Saxena. Environment-driven lexicon induction for high-level instructions. In *ACL*, 2015.
- [36] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *ICCV*, 2019.
- [37] Khanh Nguyen and Hal Daumé III. Help, Anna! Visual Navigation with Natural Multimodal Assistance via Retrospective Curiosity-Encouraging Imitation Learning. In *EMNLP*, 2019.
- [38] Khanh Nguyen, Debadeepta Dey, Chris Brockett, and Bill Dolan. Vision-based navigation with language-based assistance via imitation learning with indirect intervention. In *CVPR*, 2019.
- [39] Daniel Nyga, Subhro Roy, Rohan Paul, Daehyung Park, Mihai Pomarlan, Michael Beetz, and Nicholas Roy. Grounding robot plans from natural language instructions with incomplete world knowledge. In *CoRL*, 2018.
- [40] 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. *IJRR*, 2018.
- [41] Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. Virtualhome: Simulating household activities via programs. In *CVPR*, 2018.
- [42] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *TACL*, 2013.
- [43] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 2011.
- [44] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A platform for embodied ai research. In *ICCV*, 2019.
- [45] Ozan Sener, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Unsupervised semantic parsing of video collections. In *ICCV*, 2015.
- [46] Mohit Shridhar and David Hsu. Interactive visual grounding of referring expressions for human-robot interaction. In *RSS*, 2018.
- [47] Hao Tan, Licheng Yu, and Mohit Bansal. Learning to navigate unseen environments: Back translation with environmental dropout. In *NAACL*, 2019.
- [48] Moritz Tenorth, Daniel Nyga, and Michael Beetz. Understanding and executing instructions for everyday manipulation tasks from the world wide web. In *ICRA*, 2010.
- [49] Jesse Thomason, Daniel Gordon, and Yonatan Bisk. Shifting the baseline: Single modality performance on visual navigation & qa. In *NAACL*, 2019.
- [50] Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. Vision-and-dialog navigation. In *CoRL*, 2019.
- [51] Jesse Thomason, Shiqi Zhang, Raymond Mooney, and Peter Stone. Learning to interpret natural language commands through human-robot dialog. In *IJCAI*, 2015.
- [52] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *CVPR*, 2019.
- [53] Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *ECCV*, 2018.
- [54] Erik Wijmans, Samyak Datta, Oleksandr Maksymets, Abhishek Das, Georgia Gkioxari, Stefan Lee, Irfan Essa, Devi Parikh, and Dhruv Batra. Embodied question answering in photorealistic environments with point cloud perception. In *CVPR*, 2019.
- [55] Haonan Yu and Jeffrey Mark Siskind. Grounded language learning from video described with sentences. In *ACL*, 2013.
- [56] Licheng Yu, Xinlei Chen, Georgia Gkioxari, Mohit Bansal, Tamara L. Berg, and Dhruv Batra. Multi-target embodied question answering. In *CVPR*, 2019.
- [57] Yuke Zhu, Daniel Gordon, Eric Kolve, Dieter Fox, Li Fei-Fei, Abhinav Gupta, Roozbeh Mottaghi, and Ali Farhadi. Visual semantic planning using deep successor representations. In *ICCV*, 2017.
- [58] Dimitri Zhukov, Jean-Baptiste Alayrac, Ramazan Gokberk Cinbis, David Fouhey, Ivan Laptev, and Josef Sivic. Cross-task weakly supervised learning from instructional videos. In *CVPR*, 2019.