Hierarchical Task Learning from Language Instructions with Unified Transformers and Self-Monitoring

Yichi Zhang **Joyce Y. Chai**
Computer Science and Engineering
University of Michigan
Ann Arbor, MI, USA
{zhangyic, chaijy}@umich.edu

Abstract

Despite recent progress, learning new tasks through language instructions remains an extremely challenging problem. On the ALFRED benchmark for task learning, the published state-of-the-art system only achieves a task success rate of less than 10% in an unseen environment, compared to the human performance of over 90%. To address this issue, this paper takes a closer look at task learning. In a departure from a widely applied end-to-end architecture, we decomposed task learning into three sub-problems: sub-goal planning, scene navigation, and object manipulation; and developed a model **HiTUT**[1] (stands for **Hi**erarchical **T**asks via **U**nified **T**ransformers) that addresses each sub-problem in a unified manner to learn a hierarchical task structure. On the ALFRED benchmark, HiTUT has achieved the best performance with a remarkably higher generalization ability. In the unseen environment, HiTUT achieves over 160% performance gain in success rate compared to the previous state of the art. The explicit representation of task structures also enables an in-depth understanding of the nature of the problem and the ability of the agent, which provides insight for future benchmark development and evaluation.

1 Introduction

As physical agents (e.g., robots) start to emerge as our assistants and partners, it has become increasingly important to empower these agents with an ability to learn new tasks by following human language instructions. Many benchmarks have been developed to study the agent's ability to follow natural language instructions in various domains including navigation (Anderson et al., 2018; Chen et al., 2019), object manipulation (Misra et al., 2017; Zhu et al., 2017) and embodied reasoning (Das et al., 2018a; Gordon et al., 2018). Despite recent progress, learning new tasks through language instructions remains an extremely challenging problem as it touches upon almost every aspect of AI from perception, reasoning, to planning and actions. For example, on the ALFRED benchmark for task learning (Shridhar et al., 2020), the state-of-the-art system only achieves less than 10% task success rate in an unseen environment (Singh et al., 2020), compared to the human performance of over 90%. Most previous works apply an end-to-end neural architecture (Shridhar et al., 2020; Singh et al., 2020; Storks et al., 2021) which attempt to map language instructions and visual inputs directly to actions. While striving to top the leader board for end task performance, these models are opaque, making it difficult to understand the nature of the problem and the ability of the agent.

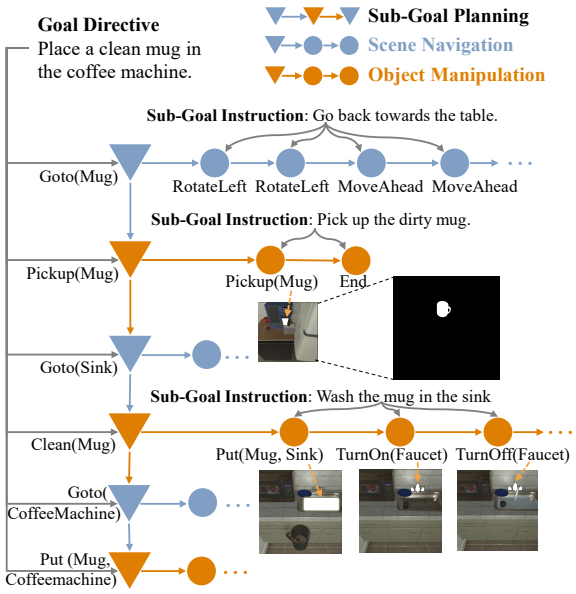


Figure 1: An example task in ALFRED.

To address this issue, this paper takes a closer look at task learning using the ALFRED benchmark. In a departure from an end-to-end ar-

[1]Source code available at https://github.com/594zyc/HiTUT

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4202–4213

chitecture, we have developed an approach to learn the hierarchical structure of task compositions from language instructions. As shown in Figure 1, a high-level goal directive ("place a clean mug in the coffee machine") can be decomposed to a sequence of sub-goals. Some sub-goals involve navigation in space (e.g., `Goto(Mug)`, `Goto(Sink)`) and others require manipulation of objects (e.g., `Pickup(Mug)`, `Clean(Mug)`). These sub-goals can be further decomposed into navigation actions such as `RotateLeft` and `MoveAhead`, and manipulation actions such as `Put(Mug, Sink)`, `TurnOn(Faucet)`. In fact, such hierarchical structure is similar to Hierarchical Task Network (HTN) widely used in AI planning (Erol et al., 1994). While this hierarchical structure is explicit and has several advantages in planning and making models transparent, how to effectively learn such structure remains a key challenge.

Motivated by recent work in multi-task learning (Liu et al., 2019a), we decomposed task learning in ALFRED into three sub-problems: sub-goal planning, scene navigation, and object manipulation; and developed a model called **HiTUT** (stands for **Hi**erarchical **T**asks via **U**nified **T**ransformers) that addresses each sub-problem in a unified manner to learn a hierarchical task structure. On the ALFRED benchmark, HiTUT has achieved the best performance with a remarkably higher generalization ability. In the unseen environment, HiTUT achieves over 160% performance gain in success rate compared to the previous state of the art.

The contributions of this work lie in the following two aspects.

An explainable model achieving the new state-of-the-art performance. By explicitly modeling a hierarchical structure, our model offers explainability and allows the agent to monitor its own behaviors during task execution (e.g., what sub-goals are completed and what to accomplish next). When a failed attempt occurs, the agent can backtrack to previous sub-goals for alternative plans to execute. This ability of self-monitoring and backtracking offers flexibility to dynamically update sub-goal planning at the inference time to cope with exceptions and new situations. It has led to a significantly higher generalization ability in unseen environments.

A de-composable platform to support more in-depth evaluation and analysis. The decomposition of task learning into sub-problems not only makes it easier for an agent to learn, but also provides a tool for an in-depth analysis of task complexity and the agent's ability. For example, one of our observations from the ALFRED benchmark is that the agent's inability to navigate is a major bottleneck in task completion. Navigation actions are harder to learn than sub-goal planning and manipulation actions. For manipulation actions, the agent can learn action types and action arguments predominantly based on sub-goals and the history of actions, while language instructions do not contribute significantly to learning. The success of manipulation actions also largely depends on the agent's ability in detecting and grounding action arguments to corresponding objects in the environment. These findings allow a better understanding of the nature of the tasks in ALFRED and provide insight to address future opportunities and challenges in task learning.

2 Related Work

Recent years have seen an increasing amount of work on in the intersection of language, vision and robotics. One line of work particularly focuses on teaching robots new tasks through demonstration and instruction (Rybski et al., 2007; Mohseni-Kabir et al., 2018). Originated in the robotics community, learning from demonstration (LfD) (Thomaz and Cakmak, 2009; Argall et al., 2009) enables robots to learn a mapping from world states to robots' manipulations based on human's demonstration of desired robot behaviors. More recent work has also explored the use of natural language and dialogue together with demonstration to teach robots new actions (Mohan and Laird, 2014; Scheutz et al., 2017; Liu et al., 2016; She and Chai, 2017; Chai et al., 2018; Gluck and Laird, 2018).

To facilitate task learning from natural language instructions, several benchmarks using simulated physical environment have been made available (Anderson et al., 2018; Misra et al., 2018; Blukis et al., 2019; Shridhar et al., 2020). In particular, the vision and language navigation (VLN) benchmark (Anderson et al., 2018) has received a lot of attention. Many models have been developed, such as the Speaker-Follower model (Fried et al., 2018), the Self-Monitoring Navigation Agent(Ma et al., 2019a; Ke et al., 2019), the Regretful Agent (Ma et al., 2019b), and the environment drop-out model (Tan et al., 2019). The VLN benchmark is further extended to study the fidelity of instruction following (Jain et al., 2019) and examined

to understand the bias of the benchmark (Zhang et al., 2020). Beyond navigation, there are also benchmarks that additionally incorporate object manipulation to broaden research on vision and language reasoning, such as embodied question answering (Das et al., 2018a; Gordon et al., 2018). The work closest to ours is the Neural Modular Control (NMC) (Das et al., 2018b), which also decomposes high-level tasks into sub-tasks and addresses each sub-task accordingly. However, self-monitoring and backtracking between sub-tasks is not explored in NMC.

The ALFRED benchmark consists of high-level goal directives such as *"place a clean mug in the coffee machine"* and low level language instructions such as *"rinse the mug in the sink"* and *"turn right and walk to the coffee machine"* to accomplish these goals. In addition to language instructions, it also comes with expert demonstrations of task execution in an interactive visual environment. We choose this dataset because its unique challenges are closer to the real world, which require the agent to not only learn to ground language to visual perception but also learn to plan for and execute actions for both navigation and object manipulation.

3 Hierarchical Tasks via Unified Transformers

As discussed in Section 1, task structures are inherently hierarchical, which compose of goals and sub-goals. Different sub-goals involve tasks of different nature. For example, navigation focuses on path planning and movement trajectories, while manipulation concerns more about interactions with concrete objects. Instead of end-to-end mapping from language instructions to primitive actions (Shridhar et al., 2020; Singh et al., 2020; Storks et al., 2021), we decomposed task learning into three separate but connected sub-problems: sub-goal planning, scene navigation, and object manipulation, and developed a model called **HiTUT** (stands for **Hi**erarchical **T**asks via **U**nified **T**ransformers) to tie these sub-problems together to form a hierarchical task structure.

3.1 Task Decomposition

We first introduce some notations to describe the task and the model. There are three types of information:

- Language ($\mathcal{L}$). We use G to denote a high-level goal directive, e.g., "place a clean mug in the coffee machine" and I_i to refer to a specific low-level language instruction.
- Vision ($\mathcal{V}$). It captures the visual representation of the environment.
- Predicates ($\mathcal{P}$). Symbolic representations are defined to capture three types of predicates: sub-goals (sg), navigation actions (a^n), and manipulation actions (a^m). Each sg has two parts (sg^{type}, sg^{arg}) where sg^{type} is the *type* (e.g., `Goto`) and sg^{arg} is the *argument* (e.g., `Knife`). Each a^n specifies a type (a^{n_type}) of action, from {`RotateLeft, RotateRight, MoveAhead, LookUp, LookDown`}. Each a^m has also two parts (a^{m_type}, a^{m_arg}) where a^{m_type} is the action type (e.g., `TurnOn`); a^{m_arg} is the action argument (e.g., `Faucet`).

Sub-Goal Planning. Sub-goal planning acquires a sequence of sub-goals $sg_1, \cdots, sg_n$ to accomplish the high-level goal G. We predict the type sg_i^{type} and argument sg_i^{arg} separately to avoid the combinatorial expansion of the output space. Previous work (Jansen, 2020) models sub-goal planning merely from high-level goal directives without visual grounding. These plans are fixed and thus not robust to potential failures during execution and variations of the visual environment. To overcome these drawbacks, our sub-goal planning is done on the fly after the previous sub-goal is executed in the environment. More specifically, our sub-goal planning objective is to learn a model (M_{sg}) that takes the visual observation at the current step (v_t), the high-level goal directive (G), and a complete sub-goal history prior to the current step ($sg_{<i}$) to predict the current sub-goal as follows:

$$sg_i \triangleq (sg_i^{type}, sg_i^{arg}) = M_{sg}(v_t, G, sg_{<i})$$

The predicted sub-goals serve as a bridge between the high-level goal and the low-level predictions of navigation actions and/or manipulation actions.

Scene Navigation. Navigation sub-goals only require predictions for the types of navigation actions. The objective is to learn a model for navigation (M_n) which takes the current visual observation (v_t), current sub-goal (sg_i), language instruction (I_i), and the navigation action history up to the current step ($a^n_{<j}$) to predict the next navigation action:

$$a_j^n \triangleq a_j^{n\_type} = M_n(v_t, I_i, sg_i, a^n_{<j})$$

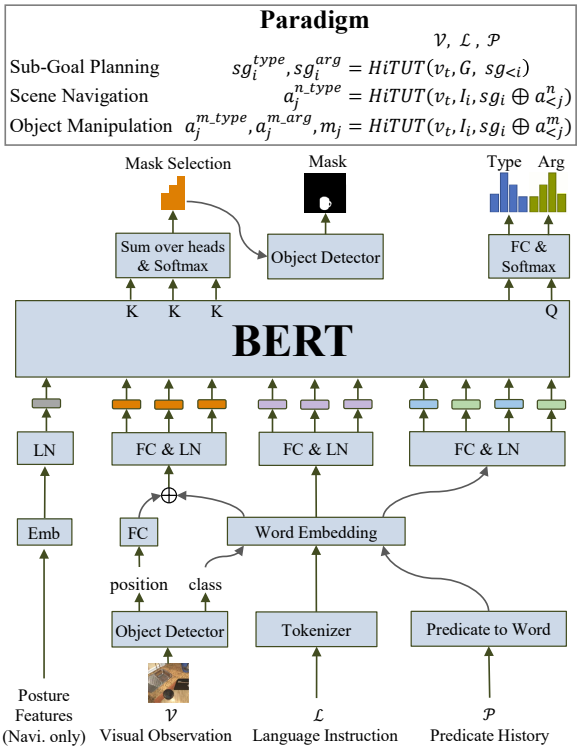


Figure 2: The structure of HiTUT.

Object Manipulation. For a manipulation sub-goal, in addition to the type and argument of the action, the model (M_m) also needs to generate a segmentation mask (m_j) on the current visual observation to indicate which object to interact with (i.e., which object the argument is grounded to):

$$(a_j^m, m_j) \triangleq (a_j^{m\_type}, a_j^{m\_arg}, m_j) = M_m(v_t, I_i, sg_i, a_{<j}^m)$$

The mask prediction is crucial because the action will not be successfully executed with an incorrect grounding even if a_j^m is correctly predicted.

As described above, although the context of the three sub-problems varies, each model has similar input components from the space of $\langle \mathcal{V}, \mathcal{L}, \mathcal{P} \rangle$. This similarity inspires us to design an unified model to solve three sub-problems simultaneously.

3.2 Unified Transformers

We leverage the effective self-attention based model (Vaswani et al., 2017) to capture the correspondence of different input sources as shown in Figure 2. We first project the input from different modalities into the language embedding space, and adopt a transformer to integrate the information together. Multiple prediction heads are constructed on top of the transformer encoder to make predictions for the sub-goal type and argument, the action type and argument, and object masks respectively. As the three sub-problems share the similar input form, we solve them all together using a unified model based on multi-task learning (Liu et al., 2019a).

Our model differs from previous works (Shridhar et al., 2020; Singh et al., 2020) in the following aspects. First, we do not apply recurrent state transitions, but feed the prediction history as the input to each subsequent prediction. This may help better capture correlations between predicates and other modalities. Second, we do not use dense visual features from the scene, but rather the object detection results. By doing this, we map different modalities to the word embedding space before feeding them into the transformer encoder, thus taking advantage of the pre-trained language models. Third, we use a predicate embedding to share linguistic knowledge between predicate symbols and word embeddings.

Predicate Embedding. We use the term predicates to refer to symbolic representations including sub-goal types, action types, and their arguments. We map symbols to their corresponding natural language phrases (e.g., `AppleSliced` is mapped to *a sliced apple*). We then tokenize and embed the tokens using word embeddings, and take the sum of the embeddings to obtain the representation of each predicate.

Vision Encoding. We use a pre-trained object detector (Mask R-CNN (He et al., 2017)) to encode visual information. Instead of dense features, we simply use the detection results (class labels, bounding box coordinates and confidence scores) as visual features. Specifically, we use the top K detected objects with a confidence score higher than 0.4 to form the visual features. The object class labels share the same space with object arguments, thus can be embedded into the same space. The position information of an object is encoded by a 7-dimensional vector consisting of its coordinates, width and height of the bounding box and its confidential score. This vector is first mapped to the same dimension as word embeddings by a liner transformation, then added to the class embedding to form the final object representation.

Object Grounding. HiTUT does not generate masks by itself. Instead it chooses an object from the K input objects and uses the corresponding mask generated by the object detector. This method makes use of the strong prior learned from object detection pre-training, so the model can focus on

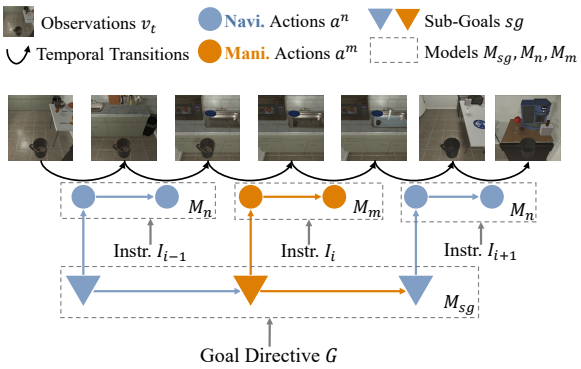


Figure 3: Overview of HiTUT where unified transformers for sub-programs are integrated together .

learning the grounding task. A drawback is that the object detector cannot be improved during training, and the performance of the detector determines the upper bound of our model's grounding ability. We leave the exploration of more robust grounding method for future work.

Posture Feature We use an additional posture feature to assist scene navigation, which includes the agent's rotation (N, S, E, W) and its angle of sight horizon (discretized by 15 degree). The positions are embedded and summed up to form the posture feature representation. The agent maintains its own posture in the form of a relative change to its initial posture instead of the absolute posture in the environment, thus avoid using additional sensory data.

3.3 Self-Monitoring and Backtracking

These unified transformers trained for sub-problems are integrated together as shown in Figure 3. One important advantage of intermediate sub-goal representations is to facilitate self-monitoring and backtracking which allows the agent to dynamically adjust the plan to cope with failures during execution. As shown in Section 4, this feature brings out the most remarkable performance gain compared to the state of the art.

Self-Monitoring. The world is full of uncertainties, and mistakes are inevitable. Based on the learned model, the agent should be able to monitor its own behaviors and dynamically update its plan when the situation arises. Our explicit representation of sub-goals allows the agent to self-check whether some sub-goals are accomplished. Particularly for manipulation sub-goals, it is feasible for the agent to detect their failures by simply monitoring whether all the manipulation actions are successfully executed. For example, `Clean(Mug)` cannot succeed if any of the actions along the path `Put(Mug, Sink), TurnOn(Facuet), TurnOff(Facuet), Pickup(Mug)` fail. When the agent detects the failure of a subgoal, for example, as shown in Figure 4 the manipulation sub-goal `Pickup(Mug)` fails, it can reason about whether the previous sub-goal (i.e., `Goto(Mug)`) is successfully achieved.

Backtracking. In classical AI, backtracking is the technique to go back and try an alternative path that can potentially lead to the goal. As shown in Figure 4, when `Pickup(Mug)` fails, the agent backtracks to `Goto(Mug)` and tries a different sequence of primitive actions to accomplish this sub-goal. In ALFRED, only based on the visual information without other sensory information (e.g., only observing a mug without knowing how far it is), is it difficult to check whether a navigation sub-goal is successfully achieved (e.g. whether a `Mug` is reachable). So every time after trying a different path for `Goto(Mug)`, the agent will check whether the subsequent manipulation action `Pickup(Mug)` is successful. If it's successful, the agent will move on to the next sub-goal; otherwise the agent will continue to backtrack until a limit on the maximum number of attempts is reached. Our explicit representation of sub-goals makes this backtracking possible and has led to a significant performance gain in unseen environments.

	Train	Validation		Test	
		Seen	Unseen	Seen	Unseen
#Scenes	108	88	4	107	8
#Demonstrations	6,574	251	255	483	488
#Annotations	21,023	820	821	1,533	1,529
#Sub-goals	162k	6.4k	6.0k	-	-
#Navi. Actions	983k	39k	35k	-	-
#Mani. Actions	209k	8.3k	8.1k	-	-

Table 1: Statistics of data distribution in ALFRED. The number of annotations is equivalent to the number of tasks in each split.

4 Experiments

4.1 Setting and Implementation

Dataset. We follow the train/validation/tests data partition proposed in ALFRED, where validation and test sets are further split into *seen* and *unseen* based on whether the scene is shown to the model during training. Each sub-goal planning step or a primitive prediction step forms a data instance for

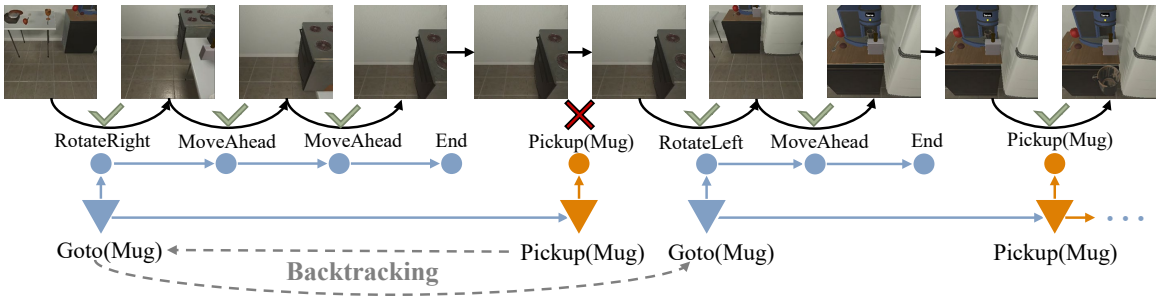


Figure 4: Illustration of self-monitoring and backtracking.

Model	Validation Seen		Validation Unseen		Test Seen		Test Unseen	
	Success	Goal-Cond	Success	Goal-Cond	Success	Goal-Cond	Success	Goal-Cond
Seq2Seq	3.70 (2.10)	10.00 (7.00)	0.00 (0.00)	6.90 (5.10)	3.98 (2.02)	9.42 (6.27)	0.39 (0.80)	7.03 (4.26)
HAM	-	-	-	-	12.40 (8.20)	20.68 (18.79)	4.50 (2.24)	12.34 (9.44)
MOCA	19.15 (**13.60**)	28.50 (**22.30**)	3.78 (2.00)	13.40 (8.30)	**22.05** (**15.10**)	28.29 (**22.05**)	5.30 (2.72)	14.28 (9.99)
HiTUT	**25.24** (12.20)	**34.85** (18.52)	**12.44** (**6.85**)	**23.71** (**11.98**)	21.27 (11.10)	**29.97** (17.41)	**13.87** (**5.86**)	**20.31** (**11.51**)
HiTUT (*G* only)	18.41 (7.59)	25.27 (12.55)	10.23(4.54)	20.71 (9.56)	13.63 (5.57)	21.11 (11.00)	11.12 (4.50)	17.89 (9.77)
Human	-	-	-	-	-	-	91.00 (85.80)	94.50 (87.60)

Table 2: Task and Goal-Condition success rates. The path length weighted version is in parentheses. The highest values per column are in **bold**. "-" denotes scores that are not reported. *G only* denotes only using the goal directive during evaluation without any sub-goal instructions.

the corresponding sub-problem. The number of data instances are shown in Table 1.

Pre-training. We employ the pre-training followed by fine-tuning paradigm for both the object detector and the main model. For the object detector, we use a Mask R-CNN (He et al., 2017) model pre-trained on MSCOCO (Lin et al., 2014), and fine-tune it on 50K images collected by replaying the expert trajectories in the ALFRED train split. As we observe that the model struggles on detecting small objects together with large receptacles, we train two networks to detect movable objects and big receptacles separately. We use the pre-trained RoBERTa (Liu et al., 2019b) model to initialize the transformer encoder.

Training. We perform imitation learning (supervised learning) on the expert demonstrations. The ground-truth labels of sub-goals and primitive actions are obtained from the metadata. Different input and output labels are organized for each sub-problem respectively as described in Section 3. We use the mask proposal that overlaps the most with the ground truth mask as the mask selection label if the intersection-of-union is above 50%. If there is no valid mask proposals, the label is assigned to 0 as an indicator of non-valid grounding. We optimize the cross-entropy loss between model predictions and the ground truth. We follow the multi-task training schema in Liu et al. (2019a) where for each iteration, a batch is randomly sampled among all the sub-problems, and the model is updated according to the corresponding objective. More details are in Appendix.

Evaluation Metrics. ALFRED leverages an interactive evaluation in the AI2-THOR environment (Kolve et al., 2017). A task is considered successful if all the goal conditions (e.g. the target object is placed on a correct receptacle and in a requested state such as heated or cleaned etc.) are met. Three measures are used: (1) success rate (the ratio of successfully completed tasks), (2) goal-condition rate (ratio of completed goal conditions), and (3) a weight version of these two rates which takes into account of the length difference between the predicted action sequence and the expert demonstrated action sequence (Shridhar et al., 2020).

Baselines. We compare HiTUT to: (1) Seq2Seq - an LSTM-based baseline model with progress monitoring proposed in Shridhar et al. (2020); (2) HAM - a hierarchical attention model over enriched visual inputs (Nguyen and Okatani, 2020), and (3) MOCA - a modular approach which also uses a Mask R-CNN for mask generation (Singh et al., 2020) and achieved previous state-of-the-art performance.

	Model	*Pick*	*Put*	*Cool*	*Heat*	*Clean*	*Slice*	*Toggle*	Avg.
Seen	Seq2Seq	32	**81**	88	85	81	25	**100**	70
	MOCA	53	62	87	84	79	51	93	73
	HiTUT	**81**	77	**95**	**100**	**83**	**81**	97	**88**
Unseen	Seq2Seq	21	46	92	89	57	12	32	50
	MOCA	44	39	38	86	71	55	11	49
	HiTUT	**71**	**69**	**100**	**97**	**91**	**78**	**58**	**81**

Table 3: Success rates of manipulation sub-goals on validation sets. The highest values per fold are in **bold**.

4.2 Evaluation Results

4.2.1 Overall Performance of HiTUT

We first evaluate the overall performance of the proposed framework as shown in Table 2. On the testing data reported by the leader board, in seen environments, HiTUT achieves comparable performance as MOCA. However in unseen environments, HiTUT outperforms MOCA by over 160% on success rate. This demonstrates our hierarchical task modeling approach has higher generalization ability compared to end-to-end models. Self-monitoring and backtracking enabled by hierarchical task structures allows the agent to better handle new situations. Remarkably, only based on high-level goal directives (i.e., *HiTUT (G Only)*) without using any sub-goal instructions, is HiTUT able to obtain a success rate of 11% in unseen environment, achieving 110% performance gain compared to MOCA. This result indicates that HiTUT can learn prior task knowledge from the hierarchical modeling process and apply that directly in new environment with some success. Nevertheless, our results are far from human performance and there is still huge room for future improvement.

To have a better understanding of the problem, we also conduct evaluations on sub-goals. The agent is positioned at the starting point of each sub-goal by following the expert demonstration and the success rate of accomplishing the sub-goal is measured. HiTUT predicts first a symbolic sub-goal representation and then the action sequence to complete the sub-goal. As shown in Table 3, HiTUT outperforms previous models on almost all of the manipulation sub-goals by a large margin. The performance gain is particularly significant in unseen environment, which demonstrates the advantage of our explicit hierarchical task modeling in low-level action planning.

	Valid Seen		Valid Unseen	
#BT	Success	Goal-Cond	Success	Goal-Cond
No	10.5 (6.0)	18.4 (13.8)	5.2 (3.0)	13.5 (11.1)
2	18.9 (9.9)	27.6 (18.0)	10.2 (5.9)	20.2 (13.6)
4	23.1 (11.3)	32.9 (18.6)	12.9 (7.0)	22.7 (12.9)
6	25.6 (12.0)	35.1 (18.5)	14.5 (7.4)	24.3 (12.3)
8	27.2 (12.5)	37.0 (18.5)	16.2 (7.8)	25.9 (12.1)

Table 4: Success rates w.r.t. the allowed maximum backtracking number (#BT).

	Seq2Seq	MOCA	Our model with different #backtracks					
			no	1	2	4	6	8
Seen	51	54	35	48	56	64	68	70
Unseen	22	32	31	45	53	60	63	65

Table 5: Success rate of the navigation sub-goal `Goto` with backtracking .

4.2.2 The Role of Backtracking

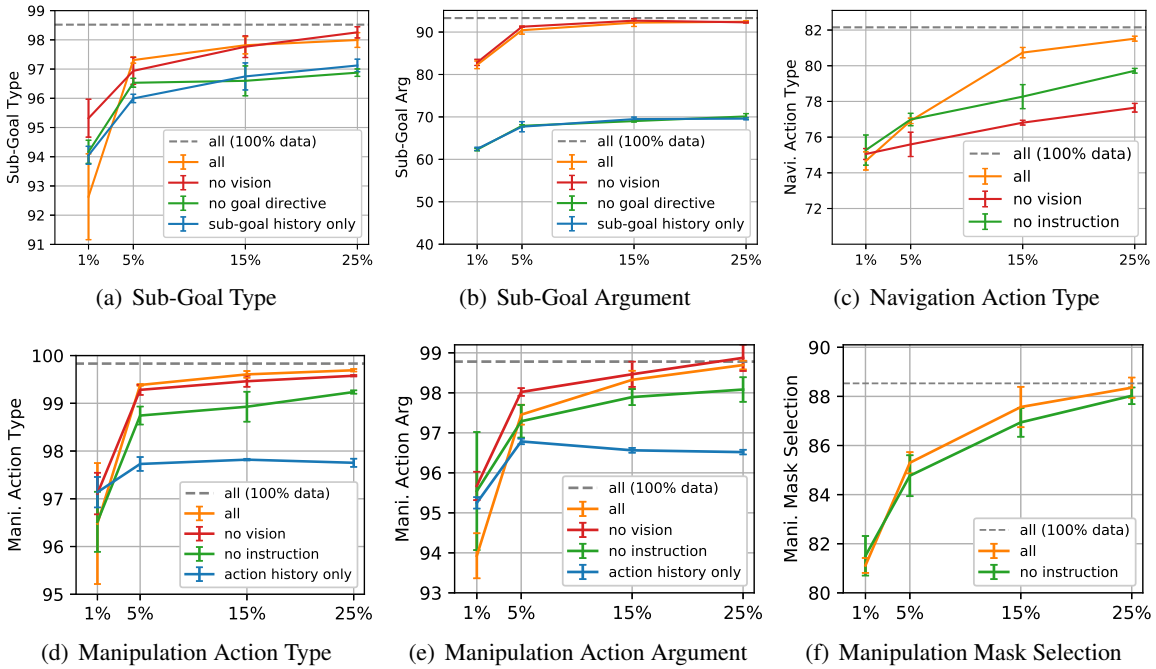
We conduct experiments to better understand the role of self-monitoring and backtracking. We repeat the task-solving evaluation with different limits on the allowed maximum number of backtracking. The agent only stops when the model predicts to stop (i.e., predicts `End`) or it reaches the backtracking limit. As shown in Table 4, as the limit increases, the task/goal-condition success rate increases accordingly. One thing notable is that the gap between success rates (weighted and unweighted) become larger when more backtrack attempts are allowed. This is within our expectation because backtracking deviates from instruction following navigation to goal-oriented exploration, which usually takes more steps than the expert demonstration.

Since backtracking is particularly targeted to navigation sub-goals `Goto` (see Section 3.3), we further examine the role of number of re-tries (i.e. backtracks) in completing the sub-goal. As shown in Table 5, HiTUT reaches more targets when given more opportunities to backtrack. The backtracking is most beneficial in unseen environment.

4.2.3 Complexity of Tasks

Task decomposition provides a tool to enable better understanding of task complexity and agent's ability. To do that, we replace different part of model predictions by the corresponding oracle sub-goals, actions, or masks, as shown in Table 6.

Using oracle sub-goals improves the success rate for 2%-6% (line SG), showing sub-goal planning is a relatively easy problem and the agent can perform reasonably well. After using the oracle



(a) Sub-Goal Type (b) Sub-Goal Argument (c) Navigation Action Type

(d) Manipulation Action Type (e) Manipulation Action Argument (f) Manipulation Mask Selection

Figure 5: Step-by-step prediction accuracies given the golden sub-goal/action history w.r.t. the proportion of training data on the unseen validation set. Each solid line corresponds to a specific input configuration. Dashed lines are the scores obtained using 100% of training data.

Method	Valid Seen		Valid Unseen	
	Success	Goal-Cond	Success	Goal-Cond
HiTUT	25.2 (12.2)	34.8 (18.5)	12.4 (6.8)	23.7 (12.0)
+ Oracle				
SG	29.0 (15.6)	39.1 (21.3)	14.0 (7.6)	25.6 (12.7)
N	75.0 (72.7)	78.0 (77.4)	57.9 (60.0)	67.7 (65.2)
SG+N	79.2 (77.8)	84.0 (81.3)	64.2 (64.2)	72.0 (68.1)
SG+N+M	89.0 (100)	90.0 (100)	80.5 (100)	83.7 (100)
SG+N+GR	99.3 (99.0)	99.4 (99.1)	99.4 (99.3)	99.6 (99.6)

Table 6: Success rates of HiTUT with different parts of predictions replaced by oracle operations with expert demonstrations. N, M, SG and GR denote oracle navigation actions, manipulation actions, sub-goals and object grounding (i.e., mask generation) respectively.

navigation actions, the seen and unseen success rates are boosted by an absolute gain of 50% and 46% respectively (line N), indicating that navigating to reach target objects is a particularly hard problem and the agent performs poorly. When oracle sub-goals, navigation actions, and manipulation actions (only symbolic representations) are given (line SG+N+M), the task success is bounded by the performance of the pre-trained object mask generator (i.e., visual grounding of the object). When oracle object masks are given together with oracle sub-goals and navigation actions (line SG+N+GR) and the agent only needs to predict symbolic representation of manipulation actions, the performance is near perfect. These last two lines indicate that predicting the type and the argument of a manipulation action is a rather simple problem in the ALFRED benchmark while grounding action arguments to the visual environment remains a challenging task.

We further examine the complexity of learning to solve sub-problems by evaluating the next-step prediction accuracy given the golden history under different conditions as shown in Figure 5. The models are trained and evaluated with different combinations of input and different amount of training data. We observe that excluding the visual input does not hurt performance for sub-goal prediction and manipulation action prediction (shown by a,b,d,e). This indicates that in ALFRED, pure symbolic planning is often independent from visual understanding, which is consistent with the findings in (Shridhar et al., 2020). However, this could be an oversimplification brought by the bias in the dataset rather than a true reflection of the physical world. For example, next action prediction can be made by remembering the correlation of predicates instead of reasoning over vision and language, due to the lack of diversity of the task environments. Removing language instructions causes a minimal performance drop of 1%-2% on action prediction tasks, which brings up the question about the usefulness of language instructions in this benchmark. Furthermore, the prediction accuracy is above 90% and 98% with only 5% training data for sub-goal

and manipulation planning respectively, while the navigation accuracy is only 82% given all the data. This again supports the finding that planning and performing navigation actions is a much harder problem than sub-goal planning and manipulation actions in ALFRED.

5 Discussion and Conclusion

This paper presents a hierarchical task learning approach that achieves the new state-of-the-art performance on the ALFRED benchmark. The task decomposition and explicit representation of subgoals enable a better understanding of the problem space as well as the current strengths and limitations. Our empirical results and analysis have shown several directions to pursue in the future. First, we need to develop more advanced component technologies integral to task learning, e.g., more advanced navigation modules through either more effective structures (Hong et al., 2020) or richer perceptions (Shen et al., 2019) to solve navigation bottleneck. We need to develop better representations and more robust and adaptive learning algorithms to support self-monitoring and backtracking. We also need to seek ways to improve visual grounding, which is crucial to both navigation and manipulation.

Second, we should also take a closer look at the construction and objective of existing benchmarks. How a benchmark is created and how truthfully it reflects the complexity of the physical world would impact the scalability and reliability of the approach in the real world. As for the objective, there is a distinction between *learning to perform tasks* and *learning to follow language instructions*. If the objective is the former, the agent should be measured by the ability to learn to accomplish high-level goal directives without being given specific language instructions at the inference time. If the objective is the latter, then the agent should be measured by how faithful it follows human instructions aside from achieving the goals, similar to (Jain et al., 2019). We need to be clear about the objectives and develop evaluation metrics accordingly.

Finally, when humans perform poorly in a complex task, we have the ability to diagnose the problem and put more energy on learning the difficult part. Physical agents should also have similar abilities. In task learning, on the one hand, the agent should be able to master simple sub-tasks from a few data instances, e.g., through a few turns of interactions with humans (Karamcheti et al., 2020). On the other hand, it should be aware of the bottleneck of its learning progress and proactively request for help when problems are encountered either during learning or during deployment (She and Chai, 2017). How to effectively design interactive and active learning algorithms for the agent to learn complex and compositional tasks remains an important open research question.

Acknowledgments

This work is supported by the National Science Foundation (IIS-1949634). The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

References

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian D. Reid, Stephen Gould, and Anton van den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 3674–3683. IEEE Computer Society.

B. D. Argall, S. Chernova, M. Veloso, and B. Browning. 2009. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483.

Valts Blukis, Yannick Terme, Eyvind Niklasson, Ross A. Knepper, and Yoav Artzi. 2019. Learning to map natural language instructions to physical quadcopter control using simulated flight. In *CoRL*.

Joyce Y. Chai, Qiaozi Gao, Lanbo She, Shaohua Yang, Sari Saba-Sadiya, and Guangyue Xu. 2018. Language to action: Towards interactive task learning with physical agents. In *IJCAI*, pages 2–9.

Howard Chen, Alane Suhr, Dipendra Misra, Noah Snavely, and Yoav Artzi. 2019. TOUCHDOWN: natural language navigation and spatial reasoning in visual street environments. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 12538–12547. Computer Vision Foundation / IEEE.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. 2018a. Embodied question answering. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 1–10. IEEE Computer Society.

Abhishek Das, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. 2018b. Neural modular control for embodied question answering. In *Conference on Robot Learning*, pages 53–62. PMLR.

Kutluhan Erol, James Hendler, and Dana S Nau. 1994. Htn planning: Complexity and expressivity. In *AAAI*, volume 94, pages 1123–1128.

Daniel Fried, Ronghang Hu, Volkan Cirik, Anna Rohrbach, Jacob Andreas, Louis-Philippe Morency, Taylor Berg-Kirkpatrick, Kate Saenko, Dan Klein, and Trevor Darrell. 2018. Speaker-follower models for vision-and-language navigation. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 3318–3329.

Kevin A Gluck and John E Laird. 2018. *Interactive task learning: Humans, robots, and agents acquiring new tasks through natural interactions.* The MIT Press.

Daniel Gordon, Aniruddha Kembhavi, Mohammad Rastegari, Joseph Redmon, Dieter Fox, and Ali Farhadi. 2018. IQA: visual question answering in interactive environments. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4089–4098. IEEE Computer Society.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. 2017. Mask R-CNN. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2980–2988. IEEE Computer Society.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez Opazo, and Stephen Gould. 2020. A recurrent vision-and-language bert for navigation. *arXiv: Computer Vision and Pattern Recognition*.

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

Peter Jansen. 2020. Visually-grounded planning without vision: Language models infer detailed plans from high-level instructions. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4412–4417, Online. Association for Computational Linguistics.

Siddharth Karamcheti, Dorsa Sadigh, and Percy Liang. 2020. Learning adaptive language interfaces through decomposition. In *Proceedings of the First Workshop on Interactive and Executable Semantic Parsing*, pages 23–33, Online. Association for Computational Linguistics.

Liyiming Ke, Xiujun Li, Yonatan Bisk, Ari Holtzman, Zhe Gan, Jingjing Liu, Jianfeng Gao, Yejin Choi, and Siddhartha Srinivasa. 2019. Tactical rewind: Self-correction via backtracking in vision-and-language navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6741–6749.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. 2017. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer.

Changsong Liu, Shaohua Yang, Sari Saba-Sadiya, Nishant Shukla, Yunzhong He, Song-Chun Zhu, and Joyce Chai. 2016. Jointly learning grounded task structures from language instruction and visual demonstration. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1482–1492, Austin, Texas. Association for Computational Linguistics.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019a. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496, Florence, Italy. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019a. Self-monitoring navigation agent via auxiliary progress estimation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Chih-Yao Ma, Zuxuan Wu, Ghassan AlRegib, Caiming Xiong, and Zsolt Kira. 2019b. The regretful agent: Heuristic-aided navigation through progress estimation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6732–6740. Computer Vision Foundation / IEEE.

Dipendra Misra, John Langford, and Yoav Artzi. 2017. Mapping instructions and visual observations to actions with reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1004–1015,

Copenhagen, Denmark. Association for Computational Linguistics.

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

Shiwali Mohan and John E. Laird. 2014. Learning goal-oriented hierarchical tasks from situated interactive instruction. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, pages 387–394. AAAI Press.

A. Mohseni-Kabir, C. Li, V. Wu, D. Miller, B. Hylak, S. Chernova, D. Berenson, C. Sidner, and C. Rich. 2018. Simultaneous learning of hierarchy and primitives (slhap) for complex robot tasks. *Autonomous Robotics*.

Van-Quang Nguyen and Takayuki Okatani. 2020. A hierarchical attention model for action learning from realistic environments and directives. *ECCV EVAL Workshop*.

P. E. Rybski, K. Yoon, J. Stolarz, and M. M. Veloso. 2007. Interactive robot task training through dialog and demonstration. In *The 2nd ACM/IEEE International Conference onHuman-Robot Interaction (HRI)*, pages 49–56.

M. Scheutz, E. Krause, B. Oosterveld, T. Frasca, and R. Platt. 2017. Spoken instruction-based one-shot object and action learning in a cognitive robotic architecture. In *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017)*, pages 1378–1386.

Lanbo She and Joyce Chai. 2017. Interactive learning of grounded verb semantics towards human-robot communication. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1634–1644, Vancouver, Canada. Association for Computational Linguistics.

William B. Shen, Danfei Xu, Yuke Zhu, Fei-Fei Li, Leonidas J. Guibas, and Silvio Savarese. 2019. Situational fusion of visual representation for visual navigation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 2881–2890. IEEE.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. 2020. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10737–10746. IEEE.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. 2020. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*.

Kunal Pratap Singh, Suvaansh Bhambri, Byeonghwi Kim, Roozbeh Mottaghi, and Jonghyun Choi. 2020. MOCA: A modular object-centric approach for interactive instruction following. *arXiv preprint arXiv:2012.03208*.

Shane Storks, Qiaozi Gao, Govind Thattai, and Gokhan Tur. 2021. Are we there yet? learning to localize in embodied instruction following. *arXiv preprint arXiv:2101.03431*.

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

A. L. Thomaz and M. Cakmak. 2009. Learning about objects with human teachers. In *Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction*, HRI '09, pages 15–22, New York, NY, USA. ACM.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yubo Zhang, Hao Tan, and Mohit Bansal. 2020. Diagnosing the environment bias in vision-and-language navigation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 890–897. ijcai.org.

Yuke Zhu, Daniel Gordon, Eric Kolve, Dieter Fox, Li Fei-Fei, Abhinav Gupta, Roozbeh Mottaghi, and Ali Farhadi. 2017. Visual semantic planning using deep successor representations. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 483–492. IEEE Computer Society.

Appendix

A Additional Training Details

We use the RoBERTa (Liu et al., 2019b) implementation from Huggingface (Wolf et al., 2019). The model is fine-tuned for 10 epochs with the Adam (Kingma and Ba, 2015) optimizer on the ALFRED training set. The learning rate warms up over the first half of the first epoch to a peak value of 1e-5, and then linearly decayed. The model achieving the highest navigation action prediction accuracy on the validation seen set is selected for evaluation. All the models are trained on one NVIDIA V100 16GB GPU.

B Additional Evaluation Details

We follow the evaluation setting in the ALFRED benchmark[2]. For each episode, the agent is given a task, which is composed of a goal directive G and several sub-goal instructions. The agent needs to sequentially perform actions to achieve the goal based on the visual observations of RGB image only. This progress ends if the agent predicts an *End* action (an *End* sub-goal for HiTUT), which is made after up to 10 failed interaction attempts or reaching the maximum step limitation. For HiTUT, there is also a maximum number of backtracking attempts, and the model will be forced to stop if the budget runs out. The maximum number of backtracking is 8 in all of our experiments without explicitly mentioning the backtracking number. We also leverage two techniques to reduce the interaction attempt failures. We use the obstruction detection trick proposed in Singh et al. (2020) to avoid failures caused by repeated tries of moving toward obstructions. We propose a self-monitoring approach to check the validity of manipulation actions. If no mask is selected or a predicted action argument is not consistent with the class prediction from Mask R-CNN for the selected object, the manipulation action is decided as failed and the agent performs a backtrack without trying to execute the action. Note that in Table 4, we remove the interaction attempt constraint when comparing the effect of different allowed maximum backtracking numbers, thus the results of $\#BT = 8$ is slightly higher than those shown in Table 2.

[2] https://leaderboard.allenai.org/alfred/submissions/get-started

Task-Type	MOCA		HiTUT	
	Seen	Unseen	Seen	Unseen
Pick & Place	29.5	5.0	**35.9**	**26.0**
Cool & Place	**26.1**	0.7	19.0	**4.6**
Stack & Place	5.2	1.8	**12.2**	**7.3**
Heat & Place	**15.8**	2.7	14.0	**11.9**
Clean & Place	22.3	2.4	**50.0**	**21.2**
Examine & Place	20.2	**13.2**	**26.6**	8.1
Pick Two & Place	11.2	1.1	**17.7**	**12.4**
Average	18.6	3.8	**25.2**	**12.4**

Table 7: Success rates across 7 task types on the validation sets. Highest values per fold are **bold**.

Model	#Backtracking	Seen SR	Unseen SR
RoBERTa	no	10.5	5.2
Scratch	no	7.9	2.8
RoBERTa	4	23.1	12.9
Scratch	4	18.1	10.2
RoBERTa	8	27.2	16.2
Scratch	8	26.8	14.0
MOCA	-	19.15	3.78

Table 8: The validation success rates for models pretrained and trained from scratch with different allowed maximum number of backtrackings.

C Additional Results

A detailed per-task performance comparison of HiTUT and MOCA is shown in Table 7. As the comparison might be unfair since HiTUT benefits from model pre-training, we also conduct an ablation study to show the effectiveness of pretraining. In Table 8, we compare the fine-tuned RoBERTa model to a Transformer with the same size trained from scratch to show the role of the RoBERTa pretraining.We can see that RoBERTa consistently improves the performance over training from scratch both w/o and w/ backtracking with an absolute gain between 0.4% and 5% on task success rate. Notably, Scratch with 4 or 8 backtrackings still outperform MOCA by a large margin in terms of the unseen success rate.