ALFRED-L: Investigating the Role of Language for Action Learning in Interactive Visual Environments

Arjun R. Akula[1]*, **Spandana Gella**[2], **Aishwarya Padmakumar**[2], **Mahdi Namazifar**[2], **Mohit Bansal**[2,3], **Jesse Thomason**[2,4], **Dilek Hakkani-Tür**[2]
[1]Google AI, [3]University of North Carolina at Chapel Hill
[2]Amazon Alexa AI, [4]University of Southern California
arjunakula@google.com, sgella@amazon.com, padmakua@amazon.com
mahdinam@amazon.com, mbansal@cs.unc.edu, jessetho@usc.edu, hakkanit@amazon.com

Abstract

Embodied Vision and Language Task Completion requires an embodied agent to interpret natural language instructions and egocentric visual observations to navigate through and interact with environments. In this work, we examine ALFRED (Shridhar et al., 2020), a challenging benchmark for embodied task completion, with the goal of gaining insight into how effectively models utilize language. We find evidence that sequence-to-sequence and transformer-based models trained on this benchmark are not sufficiently sensitive to changes in input language instructions. Next, we construct a new test split – ALFRED-L to test whether ALFRED models can generalize to task structures not seen during training that intuitively require the same types of language understanding required in ALFRED. Evaluation of existing models on ALFRED-L suggests that (a) models are overly reliant on the sequence in which objects are visited in typical ALFRED trajectories and fail to adapt to modifications of this sequence and (b) models trained with additional augmented trajectories are able to adapt relatively better to such changes in input language instructions.

1 Introduction

Recently a number of benchmark datasets have been proposed to study the ability of embodied agents to understand natural language in the context of egocentric visual observations and predict sequences of executable actions to answer questions (Das et al., 2018), navigate to desired destinations (Anderson et al., 2018; Chen et al., 2019), or additionally manipulate objects to complete tasks (Shridhar et al., 2020; Padmakumar et al., 2021).

Although multi-modal transformer-based models have achieved tremendous progress on many of these datasets (Zhu et al., 2021b; Hong et al., 2021; Pashevich et al., 2021; Zhang and Chai, 2021), analysis of such models on other visual grounding tasks has suggested that they could be learning reasoning shortcuts and exploiting unintended biases without comprehending the underlying linguistic structure (Thomason et al., 2019; Zhu et al., 2021a; Chiang et al., 2021; Akula et al., 2020; Thrush et al., 2022; Akula et al., 2021).

In this work, we analyze models trained on the ALFRED dataset (Shridhar et al., 2020) to better understand how they utilize language for embodied task completion. We chose ALFRED for our analysis as it requires object manipulation (object pick and place, opening and closing doors and more) in addition to navigation making it more challenging than most related datasets. In ALFRED, each example trajectory consists of a high-level natural language task description, followed by step-by-step (low-level) natural language instructions corresponding to logical subgoals, that when completed in sequence accomplish the task (see Fig 1).

Figure 1: An example test trajectory for task type *Pick and Place* from ALFRED. We modify the original trajectory in three different ways to create **ALFRED-L** test set (highlighted in red boxes): (a) **`NAV-ONLY`** subset picks only the navigation instructions (e.g. I1, I3, I5) to form a new trajectory; (b) **`REV-1`** subset extends the original ALFRED trajectory by adding an additional *reverse* instruction to take the agent back by one navigation step (e.g. I5-r is the reverse step formed from I5); (c) **`REV-n`** subset extends the original ALFRED trajectory by adding one or more reverse navigation steps.

* Work done in part while AA was research intern at Amazon Alexa AI in Summer 2021.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 9369–9378
December 7-11, 2022

In this work, we refer to this sequence of logical subgoals as a task structure - ALFRED trajectories consists of only 7 possible task structures. In each trajectory, an embodied agent is placed in the initial state of the environment, provided the language instructions, and expected to predict and execute a sequence of low level actions that accomplish the task described by the instructions using visual feedback from the execution of each action.

We evaluate the sensitivity of ALFRED models to changes in language instructions in two ways. We first examine whether model predictions are affected by the removal of words indicative of spatial relationships, or by the removal of step-by-step instructions entirely (§2). Our experiments demonstrate that model performance is less affected by these changes than expected. In contrast, the task completion rate of models drops to 0% when deprived of visual inputs.

In addition, we construct a new test set ALFRED-L, to test whether models can generalize to variations of ALFRED instructions that remove object manipulation steps or add navigation steps (§3). A sample from this dataset is shown in Fig 1. Intuitively, these changes do not require learning of new language understanding capabilities since following navigation instructions is already a prerequisite for successfully completing ALFRED tasks. Consequently, existing ALFRED models should be able to generalize well to such instructions. However, experimental results on ALFRED-L demonstrate that models are incapable of such generalization. We hypothesize that models overfit to the task structure of ALFRED and ignore the addition of extra objects to be visited in ALFRED-L. Further, we find that models trained using a larger number of visual scenes are able to adapt relatively better to such changes in input language instructions suggesting the importance of data augmentation techniques to make substantive progress on ALFRED.

2 Analysis by Reducing Instruction Informativeness

In this section, we examine the sensitivity of models to loss of information from language instructions when trying to complete ALFRED tasks. In our first experiment **E1**, we drop all words and phrases that indicate directional and spatial information from high and low level language instructions during inference.

We note that 81% of the tokens in ALFRED instructions constitute directional and spatial information such as *to the left*, *three steps forward*, *towards right*, and *over to the back*. We hypothesize that the absence of such crucial information should cause models to be unable to correctly navigate or identify objects being referred to.

Model	*Val-U* %	*Val-S* %	*E1-U* (Δ)%	*E1-S* (Δ)%	*E2-U* (Δ)%	*E2-S* (Δ)%
ET	2.7	31.5	(40.5)	(35.3)	(5.6)	(1.1)
ET w/o PT	2.1	26.2	(42.9)	(31.6)	(6.2)	(1.3)
ET+Synth	6.5	**44.7**	(36.9)	(21.2)	(3.8)	(0.9)
HiTuT	**12.4**	25.2	(30.6)	(29.8)	(6.7)	(4.3)
MOCA	3.7	19.1	(21.5)	(20.0)	(0.0)	(2.1)
Seq2Seq	0.0	3.7	(0.0)	(0.0)	(0.0)	(1.5)

Table 1: Task Success Rate (in percentage) of models on ALFRED validation unseen (Val-U) and seen (Val-S) splits. In perturbation **E1**, all the directional and spatial words are dropped from instructions. In perturbation **E2**, we drop all the language instructions and just keep the higher level task description. The *relative percentage drop* in success rate (shown in parentheses) before and after the perturbations is shown in the last four columns. All the numbers reported here are obtained by taking average across five experiments with different seeds.

In our second experiment **E2**, we discard all the step-by-step language instructions from the input [1] (See Appendix A for more details). We analyze the following models trained on ALFRED: (a) **Episodic Transformer (ET)**: a model that uses a transformer to encode multimodal inputs (Pashevich et al., 2021); **ET+Synth:** a version of **ET** augmented with synthetic trajectories; **ET w/o PT:** an ablated version that does not include language pretraining; (b) **HiTuT:** a hierarchical transformer-based model that explicitly predicts sub-goals in addition to low-level actions at every time step to enable backtracking to cope with execution failures (Zhang and Chai, 2021) (c) **MOCA:** a modular sequence-to-sequence model that separates action and object prediction (Singh et al., 2020); (d) **Seq2Seq:** a simple sequence-to-sequence baseline (Shridhar et al., 2020).

Table 1 shows the overall task success rates of models on validation seen and unseen splits before and after perturbations [2]. Table 2 shows the sub-goal success rates of the ET+Synth model [3]. Sub-goal success rate measures the ability of a model to accomplish the next sub-goal conditioned on the preceding ground-truth expert sequence. In perturbation **E1**, we find up to 40% relative drop in overall task success rates. Surprisingly, we see only

[1] In this setting, we re-train the model from scratch.
[2] See Appendix A for detailed results
[3] We find similar results with other models.

Sub-Goal	*Val-U* %	*Val-S* %	*E1-U* (Δ)%	*E1-S* (Δ)%	*E2-U* (Δ)%	*E2-S* (Δ)%
CleanObject	91.2	88.4	(18.5)	(16.1)	(2.5)	(1.3)
CoolObject	99.1	95.2	(−0.9)	(0.0)	(0.0)	(0.0)
GoToLoc.	50.7	80.0	(−4.3)	(1.7)	(−0.5)	(0.2)
HeatObject	99.3	94.4	(0.8)	(6.2)	(0.0)	(1.1)
PickupObject	69.0	75.9	(4.4)	(0.8)	(2.1)	(0.0)
PutObject	69.8	84.5	(6.5)	(0.9)	(1.0)	(0.0)
SliceObject	65.8	89.7	(1.3)	(5.4)	(0.0)	(1.8)
ToggleObject	83.2	98.9	(0.0)	(−1.1)	(0.0)	(0.0)

Table 2: Sub-goal Success Rate of ET+Synth model on ALFRED. The *relative percentage drop* in success rate (shown in parentheses) before and after the perturbations is shown in the last four columns. Negative percentages denote that performance of sub-goal improved after the perturbation.

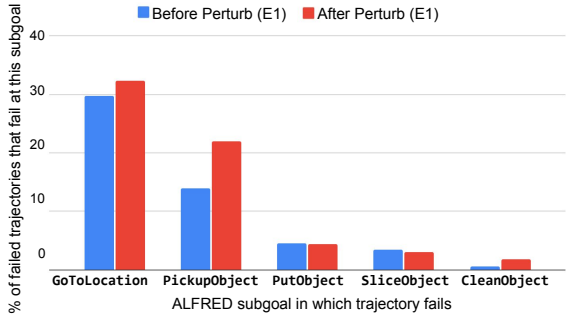

Figure 2: We examine the last subgoal of a trajectory an agent reaches before failing before and after perturbation. Most failures are in navigation (`GoToLocation`) and picking up objects (`PickUp`), which increase upon perturbation.

$< 7\%$ relative drop on all the sub-goals (except for `CleanObject`), casting doubt whether models are effectively utilizing language instructions. We explain these two contrasting observations by examining the most common sub-goal failures and low-level API action failures in overall task completion. In Fig 2, we can observe that the highest failure rate is for the `GoToLocation` sub-goal. We further examine the last incorrect action prediction that caused the failure within each subgoal and we find that most of the failures are caused by attempting to perform `PickUp` and `Put` actions within a `GoToLocation` sub-goal where an agent is only expected to perform navigation (shown in Fig 3). We also observe that dropping directional and spatial words from instructions further increases this model bias to perform `PickUp` and `Put` actions even when there is no object in view to be manipulated, leading to more failures in completing the overall task. On the other hand, with perturbation **E2**, we do not see any significant drop in model performance in both task and sub-goal success rates (Table 1; columns *E2-U(Δ)%* and *E2-S(Δ)%*). This indicates that these models fail to make effective use of the language instructions and instead exploit

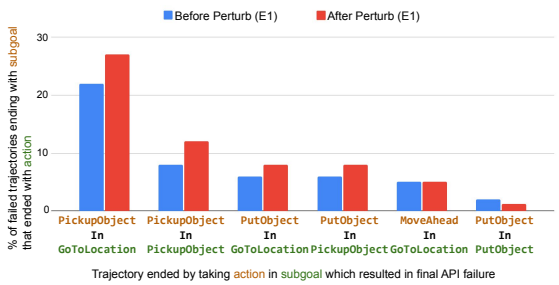

Figure 3: Most predicted trajectories fail in ALFRED due to predicting low level actions when these are infeasible. For each subgoal, we examine the percentage of trajectories that failed by last predicting particular actions. We observe that most failures are from attempting to pick up an object when the agent needs to navigate.

shallower visual correlations [4].

We additionally perform an experiment where we drop visual input instead of dropping language instructions. The accuracy drops to 0% in this case (on both val seen and unseen splits), indicating the higher influence of visual input on model performance.

3 ALFRED-L for Testing Generalization

ALFRED step-by-step language instructions involve a combination of navigation subgoals and a variety of object manipulation subgoals. Intuitively, a model that can understand such language instructions should still be able to follow modified combinations of them that add or remove some steps. To evaluate the ability of ALFRED models to generalize in this manner, we create modified instructions using examples from ALFRED validation splits. We call this modified test split **ALFRED-L**.

More concretely, training samples in ALFRED are typically a sequence of alternating navigation (`GoToLocation`) and object manipulation (for example `CleanObject`, `HeatObject`) subgoals, always ending with an object manipulation subgoal. Models would thus expect to perform object manipulations at the end of each navigation subgoal and could memorize locations of objects commonly

[4] The low-level instructions, especially in unseen environments, play an important role in navigating and identifying the objects whereas the high-level goals fail to specify the exact object locations in the environment. Therefore, if the agent is able to finish a task with just high-level goals, it could potentially imply environment-bias. For example, an object for a manipulation action could be in a different room than the agent's initial position, and it is not realistic to expect the agent to navigate, simply based on high-level goals, to a different room (which could be even far away from the initial position in real-world environments) and interact with the object to finish the task successfully.

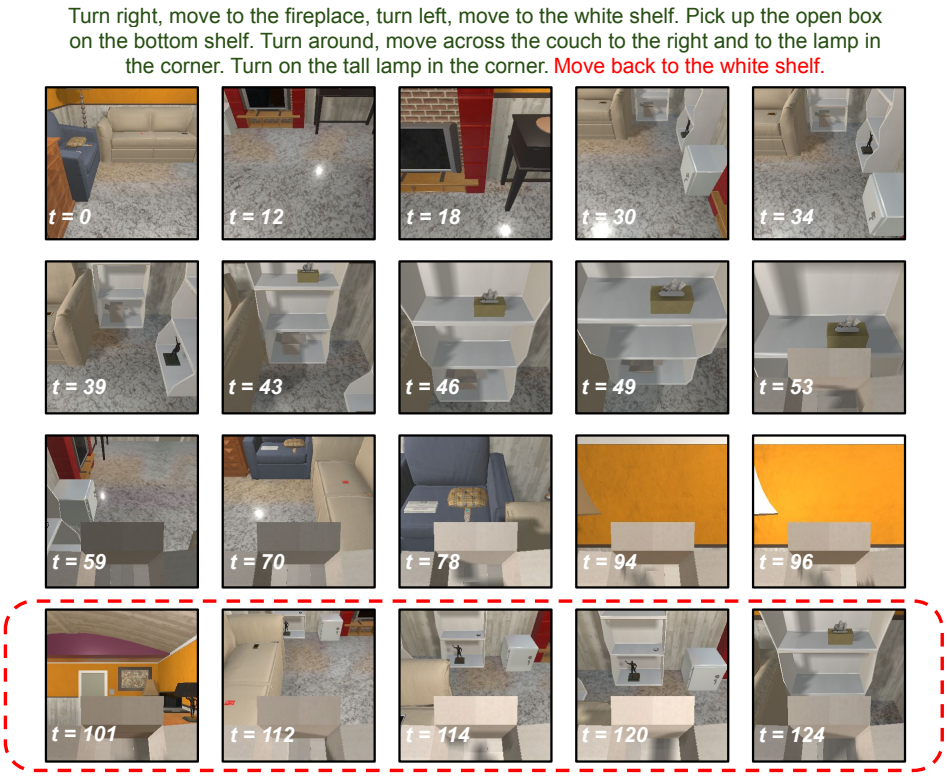

Figure 4: An example from ALFRED-L highlighting the re-generated visual frames (last row, t = 101 to 124) corresponding to the REV-1 language annotation (text at the top, highlighted in red).

navigated to, and exploit these instead of understanding instructions provided at inference time to determine objects to be navigated to. We create ALFRED-L to break some of these patterns - removing the need for object manipulation actions and adding instructions to navigate to additional objects. A model that sees a significant performance drop between ALFRED and ALFRED-L is likely overly reliant on the task structure of ALFRED and ignoring details present in language instructions.

As shown in Fig 1 and Table 3, ALFRED-L consists of 3 subsets:
(a) NAV-ONLY (Navigation-only): This is constructed by removing instructions for all object manipulation steps. An agent that understands the change made to the instructions would navigate along the same trajectory as before but without interacting with any objects.
(b) REV-1 (Reverse-1) and REV-n (Reverse-n): These add additional navigation instructions instructing the agent to backtrack to known reference positions along the trajectory. REV-1 adds one backtracking navigation step to the original ALFRED trajectory and REV-n adds more than one backtracking navigation step. These evaluate whether an agent is capable of remembering a point it had navigated to during execution so far, and navigating back to it without the expectation of performing further object manipulations [5]. In other words, using REV-1 and REV-n, our goal is to detect if the embodied agent overfit to the seen task structures as task structures in the existing unseen test splits of ALFRED only evaluate the generalization to unseen environments but fail to test generalization to unseen task structures. We expect the model to learn the capability because in collecting our language instructions for REV-1 and REV-n splits, we only leveraged the words that are used in the original ALFRED train dataset [6]. Figure 4 shows an example for the re-generated visual frames for the REV-1 instruction.

3.1 Evaluation on ALFRED-L

Table 4 shows the experimental results on ALFRED-L. Interestingly, the performance of all

[5] See Appendix B for more examples.

[6] For example, the keyword back from our REV test splits has occurred about 9793 times in ALFRED. Similarly, the REV phrases walk back, move back, go back have been used in about 1582, 396, and 1455 language instructions respectively in ALFRED.

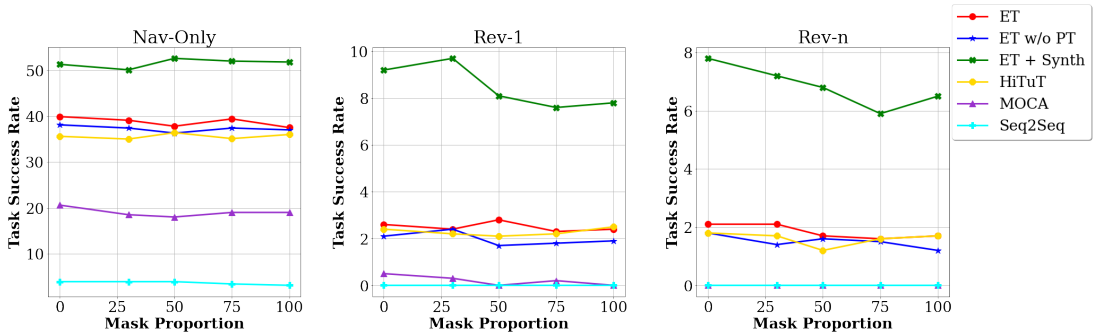

Figure 5: Performance on ALFRED-L seen splits with different proportions of masked directional and spatial words.

	ALFRED	ALFRED-L	NAV	REV-1	REV-n
Trajectories	506	1024	506	375	143
Anns.	1641	3326	1641	1219	466
Sub-goals	10710	18919	5111	9183	4625

Table 3: Statistics of ALFRED (val seen + unseen) and ALFRED-L (seen + unseen) test splits.

tested models increases by up to >30% (relative) on NAV-ONLY; whereas the performance drops by up to 91% (relative) on the REV-1 split and REV-n splits. Clearly the models fail to perform reverse navigation steps and the non-zero success rate on REV-1 and REV-n results from test samples where the model's destination in the original test set and the reverse navigation step is within the reachability threshold [7]. We hypothesize that NAV-ONLY performance is higher than that on the ALFRED test set as the agent has to perform a similar trajectory allowing the use of previously memorized knowledge about object positions but does not have to be able to frame or segment objects correctly as these do not need to be manipulated. These observations strengthen our conclusions made in section 2 that models tend to rely heavily on ALFRED task structure and visual input and ignore details present in language instructions.

We also test the models trained in section 2 on ALFRED-L. These models are trained with language instructions where different proportions of directional and spatial words are masked out. From the results of these experiments presented in Fig 5 we observe that the models do not show any sensitivity to these perturbations. Overall, **ET + Synth** is relatively more sensitive and generalizes better to ALFRED-L compared to other models, indicating that augmenting ALFRED training data with additional trajectories helps enable models to better utilize language.

[7] see Appendix B.2.

Model	ALFRED		ALFRED-L					
			NAV-ONLY		REV-1		REV-n	
	S	U	S	U	S	U	S	U
ET	31.5	2.7	39.9	11.2	2.6	0.5	2.1	0.0
ET w/o PT	26.2	2.1	38.1	8.7	2.1	0.5	1.8	0.0
ET+Synth	44.7	6.5	51.3	19.6	9.2	2.3	7.8	1.9
HiTuT	25.2	12.4	35.6	20.2	2.4	2.0	1.8	0.6
MOCA	19.1	3.7	20.6	4.5	0.5	0.0	0.0	0.0
Seq2Seq	3.7	0.0	3.9	0.0	0.0	0.0	0.0	0.0

Table 4: Comparison of model performance (Task Success Rate in percentage) on Validation Seen (S), Unseen (U) splits between ALFRED and ALFRED-L splits.

4 Conclusion

We evaluate embodied task completion models trained on ALFRED and find that they are not very sensitive to loss of spatial and directional information, or detailed task steps. We also present a new test split ALFRED-L to test generalization to novel task structures and find that models are unable to adapt to the addition of extra reverse navigation steps.

We hope that our work guide the development of future embodied AI benchmarks (and models) to avoid the issues we identified with ALFRED. In addition, our analysis at the sub-goal level on unseen environments (unseen test) and on our proposed ALFRED-L test split help test different generalization aspects and therefore can potentially represent major failure modes in other embodied AI datasets.

5 Limitations

We analyze ALFRED models for their sensitivity to modifications in input language instructions. However since our analysis is restricted to dataset in English, we are unsure whether similar behavior will be observed in languages other than English for embodied task completion. We hypothesize that such behavior will be less likely in datasets spanning more tasks or where less of the scene is visible from any given position the agent is in.

Additionally we do not include models that make semantic maps of the environment in this analysis. However, some such works (Blukis et al., 2022; Min et al., 2021) have stated that the difference in performance when step-by-step instructions are removed is low.

Another limitation of our work is that unlike in the creation of ALFRED where each trajectory is annotated with 3 (or more) sets of language instructions, in ALFRED-L we only provide a single language instruction for the additional reverse navigation steps added in `REV-1` and `REV-n` splits. Since the examples in `REV-1` and `REV-n` include all the steps of the original ALFRED trajectory, we rely on the diversity between the original sets of ALFRED instructions to separate the resultant instructions in ALFRED-L.

References

Arjun Akula, Spandana Gella, Yaser Al-Onaizan, Song-chun Zhu, and Siva Reddy. 2020. Words aren't enough, their order matters: On the robustness of grounding visual referring expressions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6555–6565.

Arjun Akula, Spandana Gella, Keze Wang, Song-chun Zhu, and Siva Reddy. 2021. Mind the context: the impact of contextualization in neural module networks for grounding visual referring expressions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6398–6416.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. 2018. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3674–3683.

Valts Blukis, Chris Paxton, Dieter Fox, Animesh Garg, and Yoav Artzi. 2022. A persistent spatial semantic representation for high-level natural language instruction execution. In *Conference on Robot Learning*, pages 706–717. PMLR.

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

Ting-Rui Chiang, Yi-Ting Yeh, Ta-Chung Chi, and Yau-Shian Wang. 2021. Are You Doing What I Say? On Modalities Alignment in ALFRED. In *Novel Ideas in Learning-to-Learn through Interaction Workshop at EMNLP 2021*.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, Devi Parikh, and Dhruv Batra. 2018. Embodied question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–10.

Yicong Hong, Qi Wu, Yuankai Qi, Cristian Rodriguez-Opazo, and Stephen Gould. 2021. Vln bert: A recurrent vision-and-language bert for navigation. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1643–1653.

So Yeon Min, Devendra Singh Chaplot, Pradeep Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. 2021. Film: Following instructions in language with modular methods. *arXiv preprint arXiv:2110.07342*.

Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gokhan Tur, and Dilek Hakkani-Tur. 2021. TEACh: Task-driven Embodied Agents that Chat. *arXiv preprint arXiv:2110.00534*.

Alexander Pashevich, Cordelia Schmid, and Chen Sun. 2021. Episodic transformer for vision-and-language navigation. *arXiv preprint arXiv:2105.06453*.

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. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10737–10746.

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*.

Jesse Thomason, Daniel Gordon, and Yonatan Bisk. 2019. Shifting the baseline: Single modality performance on visual navigation & qa. In *NAACL*.

Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. 2022. Winoground: Probing vision and language models for visio-linguistic compositionality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5238–5248.

Yichi Zhang and Joyce Yue Chai. 2021. Hierarchical task learning from language instructions with unified transformers and self-monitoring. *ArXiv*, abs/2106.03427.

Wanrong Zhu, Yuankai Qi, P. Narayana, Kazoo Sone, Sugato Basu, Xin Eric Wang, Qi Wu, Miguel P. Eckstein, and William Yang Wang. 2021a. Diagnosing vision-and-language navigation: What really matters. *ArXiv*, abs/2103.16561.

Wanrong Zhu, Xin Wang, Tsu-Jui Fu, An Yan, P. Narayana, Kazoo Sone, Sugato Basu, and William Yang Wang. 2021b. Multimodal text style transfer for outdoor vision-and-language navigation. In *EACL*.

Appendix

In this supplementary material, we begin by providing more details on our perturbation experiments to supplement Section 2 of the main paper. We then present additional details on our ALFRED-L annotation, and show a few examples randomly sampled from ALFRED-L to supplement Section 3.

A Reducing Instruction Informativeness

As discussed in Section 2 of the main paper, in perturbation experiment **E1**, we drop all directional and spatial words from both high and low level language instructions. Note that, in **E1**, we only perturb samples during inference and use original unperturbed data during training. In Figure 6 we present a examples of these perturbations. On the other hand, in experiment **E2**, we discard all low level language instructions during both training and inference, and do not perform any perturbation on high level language instructions. We closely followed the original set up used by the ET and other models proposed for ALFRED dataset (batch size, learning rate, pre-training, iterations, etc) for training and inference. All these models are trained using 4 to 8 NVIDIA A100 and V100 instances. Table 5 and Table 6 present the absolute percentage of task and sub-goal success rate of the models in **E1** and **E2** settings - corresponding to Table 1 and Table 2 in Section 2 of the main paper.

B ALFRED-L Test Splits

As discussed in Section 3 of the main paper, we construct ALFRED-L test split by performing modifications to the trajectories from ALFRED validation Seen and Unseen splits. In the below subsections we provide the details on the creation of the ALFRED-L splits and the modifications made to the trajectories.

B.1 `NAV-ONLY` subset

This is constructed by removing language instructions for all object manipulation steps. After deleting all the interaction sub-goals, we re-generate the visual scenes using *render_trajs.py* script from `https://github.com/alexpashevich/E.T./tree/master/alfred/gen` - to make the visual inputs to be consistent with the language inputs.

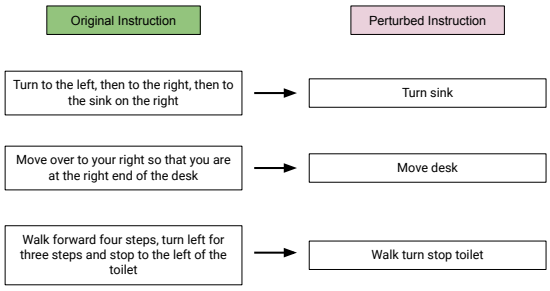

Figure 6: Examples for experiment **E1** perturbations

Model	*Val-U* %	*Val-S* %	*E1-U* %	*E1-S* %	*E2-U* %	*E2-S* %
ET	2.7	31.5	1.6	20.3	2.5	31.1
ET w/o PT	2.1	26.2	1.2	17.9	2.0	25.8
ET+Synth	6.5	**44.7**	4.1	35.2	6.2	44.1
HiTuT	**12.4**	25.2	8.6	17.6	11.5	24.1
MOCA	3.7	19.1	2.9	15.2	3.7	18.7
Seq2Seq	0.0	3.7	0.0	3.7	0.0	3.6

Table 5: Task Success Rate (in percentage) of models on ALFRED validation unseen (Val-U) and seen (Val-S) splits in **E1** and **E2**.

For example, consider a sample json structure for the trajectories in ALFRED as shown in Figure 7. To create `NAV-ONLY` trajectory from this json,
(a) We first collect the high-level indices (`high_idx`) of all the `GoToLocation` sub-goals using the json object `plan -> high_pddl`.
(b) We next filter out the navigation low-level actions indices (i.e. `low_idx` in `low_actions` json object) for the corresponding navigation high-level indices filtered in previous step.
(c) Next, we remove all the images from the `images` json object which does not contain the selected `high_idx` and `low_idx` values.
(d) We then remove all the language annotations for the manipulation actions in the `turk_annotations -> anns -> high_descs` based on the selected `high_idx`. Note that high-level indices (`high_idx`) has one-to-one mapping with the language annotations.
(e) We pass this updated json to the *render_traj.py* script to re-generate the images.

Table 7 and Table 8 show few examples of the language annotations in the original trajectory and in the modified `NAV-ONLY` trajectory. Note that we explicitly capture the final expected position of the agent and modify the original ALFRED evalua-

Sub-Goal	*Val-U* %	*Val-S* %	*E1-U* %	*E1-S* %	*E2-U* %	*E2-S* %
CleanObject	91.2	88.4	74.3	74.1	88.9	87.2
CoolObject	99.1	95.2	99.9	95.2	99.1	95.2
GoToLocation	50.7	80.0	52.8	78.6	50.9	79.8
HeatObject	99.3	94.4	98.5	88.5	99.3	93.3
PickupObject	69.0	75.9	65.9	75.2	67.5	75.9
PutObject	69.8	84.5	65.2	83.7	69.1	84.5
SliceObject	65.8	89.7	64.9	84.8	65.8	88.0
ToggleObject	83.2	98.9	83.2	99.9	83.2	98.9

Table 6: Sub-goal Success Rate of ET+Synth model on ALFRED ALFRED validation unseen (Val-U) and seen (Val-S) splits in **E1** and **E2**.

tion pipeline to only consider this final position for computing task success rate. While evaluating model performance on NAV-ONLY split, if the agent is within 5 steps (i.e. `reachability threshold` $\leq$ 1.25, where the step size in AI2Thor is 0.25) away from the ground-truth destination, we consider the task to be successful.

B.2 REV-1 and REV-n splits

These add additional navigation instructions instructing the agent to backtrack to known reference positions along the trajectory. **REV-1** adds exactly one navigation step to the original set of step-by-step instructions from ALFRED and **REV-n** adds more than one navigation steps. The authors of this work annotate the language instructions for these reverse steps. We perform multiple validation steps and delete the trajectories that are ambiguous or not clear. Table 7 and Table 8 show few examples of the language annotations in the original trajectory and in the modified REV-1 and REV-n trajectories. In re-generating the visual scenes for the newly added reverse instructions, we first add new sequence of low-level actions to the json structure by reversing the order of original navigation steps and then pass the updated json structure to the https://github.com/alexpashevich/E.T./tree/master/alfred/gen/render_traj.py script to generate corresponding images.

For example, if the original sequence of navigation contains low-level actions such as MoveForward → MoveForward → PickUp → RotateLeft → MoveForward → Look Down, the reversed navigation actions would be LookUp → MoveForward → RotateRight → MoveForward → MoveForward. As we can see, we interchange RotateRight and RotateLeft; LookUp and LookDown actions while backtracking. Also,

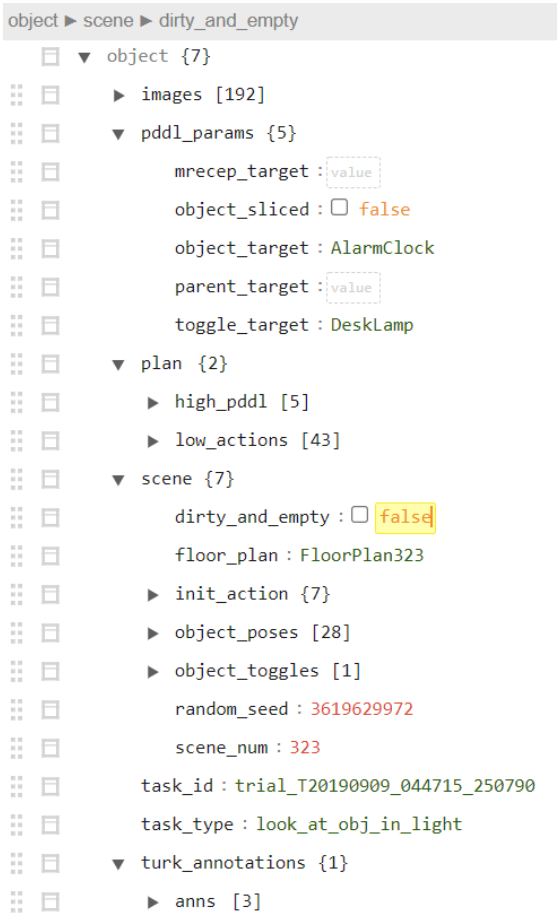

Figure 7: JSON structure for the trajectories in ALFRED

the first time we initiate the reverse backtracking, we perform RotateRight action twice. Moreover, all the object interaction actions such as PickUp are skipped while backtracking.

Similar to NAV-ONLY trajectories, we explicitly capture the final expected position of the agent and modify the original ALFRED evaluation pipeline to consider final position of the agent in addition to the object interaction tasks, for computing task success rate. We set the `reachability threshold` to be ≤ 1.25, where the step size in AI2Thor is 0.25. In Table 4 of the main paper, we find the performances of all the models on REV-1 and REV-n drops to 0 on both seen and unseen splits when we decrease the `reachability threshold` to ≤ 0.5.

	Task Type: *pick clean then place in recep* **High-level Goal:** Put a knife in the sink before standing it on the counter.
Original Instructions	Go to the right and walk to the fridge, hang a right and go to the counter between the dishwasher and stove. Pick up the potato that is on the counter. Go right to the microwave. Put the potato in the microwave, turn it on to cook, remove the potato. Go left towards toward the fridge, then hang a left, go to the garbage can. Put the potato in the garbage can.
`NAV-ONLY` Instructions	Go to the right and walk to the fridge, hang a right and go to the counter between the dishwasher and stove. Go right to the microwave. Go left towards toward the fridge, then hang a left, go to the garbage can.
`REV-1` Instructions	Go to the right and walk to the fridge, hang a right and go to the counter between the dishwasher and stove. Pick up the potato that is on the counter. Go right to the microwave. Put the potato in the microwave, turn it on to cook, remove the potato. Go left towards toward the fridge, then hang a left, go to the garbage can. Put the potato in the garbage can. Walk back to the microwave.
`REV-n` Instructions	Go to the right and walk to the fridge, hang a right and go to the counter between the dishwasher and stove. Pick up the potato that is on the counter. Go right to the microwave. Put the potato in the microwave, turn it on to cook, remove the potato. Go left towards toward the fridge, then hang a left, go to the garbage can. Put the potato in the garbage can. Walk back to the microwave. Return to the counter between the dishwasher and stove.

Table 7: Random example from ALFRED-L. We show original crowd-sourced instructions from ALFRED as well as our modified ALFRED-L instructions in `NAV-ONLY`, `REV-1` and `REV-n` setting.

	Task Type: *pick clean then place in recep* **High-level Goal:** Clean a knife and put it back onto the counter.
Original Instructions	Turn left and move to the gray coffee maker to the right of the lettuce, then move to the silver dishwasher to the right of the black toaster. Pick up the yellow handled knife to the left of the square plate from the counter. Turn around and move to the sink to the right of the loaf of bread. Place the knife in the sink to the left of the lettuce, turn on the faucet to rinse the knife, then pick up the knife from the sink. Turn around and face the dishwasher underneath the green glass. Place the knife on the plate to the rear of the potato on the counter.
NAV-ONLY Instructions	Turn left and move to the gray coffee maker to the right of the lettuce, then move to the silver dishwasher to the right of the black toaster. Turn around and move to the sink to the right of the loaf of bread. Turn around and face the dishwasher underneath the green glass.
REV-1 Instructions	Turn left and move to the gray coffee maker to the right of the lettuce, then move to the silver dishwasher to the right of the black toaster. Pick up the yellow handled knife to the left of the square plate from the counter. Turn around and move to the sink to the right of the loaf of bread. Place the knife in the sink to the left of the lettuce, turn on the faucet to rinse the knife, then pick up the knife from the sink. Turn around and face the dishwasher underneath the green glass. Place the knife on the plate to the rear of the potato on the counter. Walk back to the sink to the right of the loaf of bread.
REV-n Instructions	Turn left and move to the gray coffee maker to the right of the lettuce, then move to the silver dishwasher to the right of the black toaster. Pick up the yellow handled knife to the left of the square plate from the counter. Turn around and move to the sink to the right of the loaf of bread. Place the knife in the sink to the left of the lettuce, turn on the faucet to rinse the knife, then pick up the knife from the sink. Turn around and face the dishwasher underneath the green glass. Place the knife on the plate to the rear of the potato on the counter. Walk back to the sink to the right of the loaf of bread. Move to the silver dishwasher to the right of the black toaster.

Table 8: Random example from ALFRED-L. We show original crowd-sourced instructions from ALFRED as well as our modified ALFRED-L instructions in NAV-ONLY, REV-1 and REV-n setting.