11-777 Spring 2021 Class Project

Andrew Singh* **Ankit Ramchandani*** **Vashisth Parekh***
{andrewsi, aramchan, vparekh}@andrew.cmu.edu

Abstract

Template for 11-777 Reports using the ACL 2021 Style File

1 Introduction and Problem Definition (1-1.25 pages)

Thesis statement or Hypothesis we are aiming to prove
"Our approach is better is not a hypothesis"

*Everyone Contributed Equally – Alphabetical order

2 Related Work and Background

2.1 ALFRED

Though ALFRED is a relatively new task, a multitude of approaches have recently been proposed that demonstrate improved performance over the baseline introduced in (Shridhar et al., 2020a). The baseline model consists of a CNN to encode the visual input at each timestep, a bi-LSTM to encode the language directives, and a decoder LSTM to infer the action at each timestep while attending over the language encoding. (Corona et al., 2020) used the same architecture as the baseline, except they maintained separate modules for each *subgoal type* (e.g., GOTO, PICKUP). They then used a high-level controller to choose which module to execute at each step based on the language directives. (Singh et al., 2020) proposed a vision module for generating interaction masks and an action module for predicting actions, with the vision module first predicting the class of the object of interest and then generating the pixel-wise interaction mask given the predicted object class. (Storks et al., 2021) addressed ALFRED's long action sequences by training the model to execute one subgoal at a time rather than all subgoals at once, and they address the agent's poor navigation performance by augmenting the agent's perception with additional viewing angles that are used for training an object detection module and for predicting the agent's orientation angle relative to the goal.

Unlike methods that learn a direct mapping from observations to actions end-to-end, (Saha et al., 2021) proposed a truly modular framework that is able to learn from unaligned or weakly aligned data as opposed to requiring expert demonstrations. Their mapping module includes a novel mapping scheme based on graph convolutional networks (Kipf and Welling, 2016) for improved navigation, and their language module leverages a pre-trained model to perform joint intent detection and slot filling on the language directives.

While the previously mentioned works improve upon the modeling approach proposed in the original ALFRED paper, (Shridhar et al., 2020b) also proposed a new environment to address the challenge of generalizing to unseen tasks. They aligned tasks in ALFRED with a purely textual environment, TextWorld (Côté et al., 2018), allowing agents to first learn in an abstract setting in order to generalize better in the embodied setting. They additionally introduced a modular architecture to demonstrate the effectiveness of their ALFWorld environment, consisting of a state estimator that translates visual observations to text, an abstract text agent pre-trained in TextWorld that generates high-level actions from textual observations and a goal, and a controller that translates high-level actions to sequences of low-level actions in the embodied environment.

To the best of our knowledge, the papers discussed above cover all of the approaches proposed thus far that attempt the full ALFRED task.

2.2 Related Tasks

2.2.1 Vision and Language Navigation

While ALFRED requires navigation and interaction with objects based on visual and language input, a related task is just vision and language based navigation (VLN). (Fried et al., 2018) proposed a policy consisting of two modules: an instruction follower model that produced a step-by-step action sequence from visual and textual input, and a speaker model that predicted the probability that a particular language instruction describes a given sequence. To predict the final trajectory, multiple trajectories were first generated by the follower model, and the one most likely to match with the natural language description (as assessed by speaker model) was chosen. (Wang et al., 2019) proposed an improvement over the previous method by learning a LSTM (Hochreiter and Schmidhuber, 1997) and attention-based policy using reinforcement learning (RL). They also used a "speaker model" which was used to generate an intrinsic reward for the RL algorithm based on the probability of the language description matching the predicted action sequence. (Wang et al., 2018) proposed a method to jointly perform imitation and reinforcement learning using a policy that consisted of an "action predictor" used to predict the final action based on inputs coming from model-free and model-based RL modules. The model-free module was an LSTM and attention-based network similar to other approaches (Shridhar et al., 2020a; Wang et al., 2019). The model-based module "imagined" multiple different trajectories in the future, and produced a combined representation to the action predictor. (Wani et al., 2020) performed several experiments on a long-horizon navigation task in a realistic 3D setting to empirically show that using a semantic map-like memory can significantly boost navigation performance. (Hao et al., 2020) pre-

trained their model on image-text-action triplets in a self-supervised manner. Their model was able to generalize better in unseen environments, improving the SOTA in the Room-to-Room task (similar to ALFRED). (Majumdar et al., 2020) improved VLN performance by using a visiolinguistic transformer based model that scores the compatibility between an instruction and a particular visual scene. Pretraining on the image-text pairs from the web improved the performance of the VLN.

2.2.2 Embodied Question Answering

Embodied Question Answering (Das et al., 2018a) (EmbodiedQA) is a related task in which an agent spawns at a random location in a 3D environment and is asked a question about an object. To correctly answer, the agent must navigate the environment and gather information through egocentric vision about the object and its surroundings. This challenging task requires many of the skills needed in the ALFRED benchmark, including active perception, commonsense reasoning, goal-driven navigation, and grounding language to vision and actions.

(Das et al., 2018a) proposed an approach with a two-step navigation module: a planner that selects actions and a controller that executes those actions a variable number of times. Their agent is initialized via imitation learning and then fine-tuned via reinforcement learning. (Das et al., 2018b) improved upon this approach by introducing a high-level policy that proposes compositional sub-goals to be executed by sub-policies. They train their model via imitation learning in a bottom-up fashion, first training the sub-policies before training the high-level policy. They then fine-tune their model via reinforcement learning in a similar bottom-up fashion, allowing the high-level policy to adapt to the behavior of the sub-policies.

(Yu et al., 2019) generalized the EmbodiedQA task to multiple targets; instead of a question asking only about a single object, it may ask about several objects and require comparative reasoning. They propose a novel architecture for the task consisting of four modules: a program generator that converts the question to executable sub-programs, a navigator that executes these sub-programs to guide the agent to relevant locations, a controller that selects relevant observations along the agent's path, and a visual question answering module that uses the observations from the controller to predict the final answer.

2.3 Relevant ML Methods

2.3.1 Multimodal Alignment

In ALFRED, the agent receives all natural language instructions at the beginning of the episode, but receives visual observations at each time-step. It is imperative for the agent to align the natural language directives with its current visual observation so that it can spot objects of interest described in natural language in the current visual scene. In this section, we summarize some research in multimodal machine learning focused on this problem of learning such soft alignment (Baltrušaitis et al., 2018).

(Yu and Ballard, 2004) used a graphical model to align objects in (egocentric) images with spoken words. (Mei et al., 2015) uses a bi-LSTM with a multi-level aligner to map instructions with navigational actions. (Ma et al., 2019) proposed a visual textual co-grounding alignment mechanism and a corresponding progress monitor. They used the hidden state from the previous timestep of their LSTM to generate textual and visual grounding, which helps their agent decide which action to take next. Similarly, (Wang et al., 2019) used an LSTM to predict actions and included an attention mechanism on visual and textual input based on the current hidden state of the LSTM. (Ke et al., 2019) used attention mechanism over language to compute how the previous action aligned with the description. (Wang et al., 2018; Shridhar et al., 2020a) both used modules which perform attention over textual input using the hidden state of the LSTM, so that the agent knows which words in the input text to focus on.

2.3.2 Generalization in Multimodal Settings

The ALFRED dataset has unseen splits of validation and test data which measure generalization of the learned policy, but multimodal models are more prone to overfitting due to their increased capacity (Wang et al., 2020). To this end, (Wang et al., 2020) also proposed a gradient blending approach, which computes optimal blends of modalities based on overfitting behavior. This approach achieves SOTA results on egocentric action task similar to ALFRED. (Alet et al., 2019) presented a meta-learning strategy where they separately trained each modular component on related tasks and then combined them to create a more general model that scales across tasks. More specifically to ALFRED, (Nguyen and Okatani, 2018) proposed multi-task

learning approach that enables visual-language representations that can be generalized to other tasks. Their algorithm used representation encoders that learn hierarchical features by fusing visual and semantic representations and task specific decoders that decode those features however they see best fit for the given task.

2.3.3 Reinforcement and Imitation Learning

All known SOTA approaches (Singh et al., 2020; Corona et al., 2020; Storks et al., 2021) for ALFRED use imitation learning (IL) (Hussein et al., 2017), despite IL having several known limitations because the standard i.i.d assumptions are not met (Ross and Bagnell, 2010). Methods like DAgger (Ross et al., 2011) that attempt to mitigate the limitations of IL cannot be applied directly because new data cannot be generated on the fly in ALFRED (Shridhar et al., 2020a). For these reasons, in this section, we describe methods that use a combination of IL and reinforcement learning (RL) techniques in addition to the ones that were covered in Section 2.2.1 (Wang et al., 2019, 2018).

Many methods have been proposed to use RL methods when expert data is present. (Ho and Ermon, 2016) proposed a method to directly learn a policy from expert data that optimizes a reward function that would be obtained by running inverse RL (Abbeel and Ng, 2010) on expert data. Notably, their method directly outputs the policy and does not involve running inverse RL to extract the reward function first, which could be very costly. They experimentally show that their method outperforms other IL methods and often achieves expert level performance. (Reddy et al., 2019) proposed a much simpler alternative to (Ho and Ermon, 2016) which still achieves competitive performance. They simply give the agent a positive reward when it matches the expert action and no reward otherwise. This simple idea is theoretically motivated and forces the agent to return to states seen by the expert. (Salimans and Chen, 2018) proposed a RL-based method to solve the challenging Atari game, Montezuma's Revenge, using a single demonstration. Their main contribution was to train the agent using a curriculum: they trained the RL agent to reach the goal by starting from states in the demonstration in reverse order. In other words, they first trained an RL agent to reach the goal state from second-to-last state in the demonstration, then from third-to-last, and so on. The main insight was that the RL agent had to learn only a sub-task at each step, overcoming any hard exploration. (Hester et al., 2018) proposed a method to do Deep Q-Learning (Mnih et al., 2013) from demonstrations by adding a term to the loss function that forces the Q-value of the expert action to be at least a margin higher than other actions. This term adds a trade-off between following the expert action and the optimal action as predicted by the Q-values. (Rajeswaran et al., 2017) proposed a method to learn complex, non-trivial manipulation tasks using RL and IL. They use IL to warm start the policy, and then train it using RL with a modified gradient update which forces the policy to stay close to expert actions. (Garmulewicz et al., 2018) proposed a simple modification to the loss function used in actor-critic methods to account for expert data and showed that their simple modification can achieve satisfactory results on challenging tasks like Montezuma's Revenge. (Nair et al., 2018) also proposed a method that involves a modification to the loss function to account for IL, but they only add this extra loss term when the learned critic believes that the expert actions are indeed better than the policy action. In other words, their modification accounts for cases when expert data may not be perfect.

3 Task Setup and Data

3.1 Task Definition

We plan to work with the ALFRED dataset (Shridhar et al., 2020a) with the goal of learning a set of actions in an indoor household setting which will help an agent complete a task described by natural language. The tasks require navigation and interaction with multiple objects in the scene. Each interaction action requires a pixel-wise interaction mask to specify the object of interest. The agent receives high-level and low-level natural language instructions at the beginning of the episode, and can use egocentric visual observation (i.e. access to current RGB image, depth map, and instance segmentation map) at each time step as input. The agent produces one or two outputs at each time step: the current action to take, and, if the action involves interaction, the interaction mask of an object of interest.

We intend to predict the interaction mask pixel-wise instead of using any other coarser representations like bounding boxes. We also intend to use inputs in their rawest representation (e.g. raw image data instead of extracted features) for maximum generality and flexibility of downstream methods. Furthermore, we plan to develop a method to solve the full task of navigation and interaction in the ALFRED dataset. We clarify this to convey that we are not working with a small sub-task or a sub-problem of the dataset. Since current methods (Corona et al., 2020; Singh et al., 2020; Shridhar et al., 2020a) struggle with generalization to novel objects and environments, we will attempt to primarily focus on improving generalization performance, which is measured by an "unseen" split of the test set which contains new environments and objects.

3.2 Dataset Statistics

In this section, we present a subset of the analysis we performed. **We encourage the reader to see the Jupyter notebook stored in the *Analysis* folder for full list of figures pertaining to the analysis since this report only includes a subset.**

3.2.1 Metadata Analysis

In this section, we analyze various metadata in ALFRED. Table 1 shows average values for the quantitative metrics we chose to measure on ALFRED. We can see that the averages are fairly consistent across splits. The navigation-interaction ratio indicates that for every interaction action in a demonstration, there are roughly 9 navigation actions. The mask coverage indicates that on average, the ground-truth interaction mask covers a rather small (15-17%) proportion of the image. The step-object coverage of nearly 1 indicates that for almost all interactions, the name of the object of interest is mentioned in the corresponding language directive. By manually inspecting examples with low interaction step coverage, we find that the object's name are usually substituted with a synonym (e.g."rag" for "cloth" and "scoop" for "ladle").

	Train	Valid (seen)	Valid (unseen)
Steps per directive	6.68	6.64	6.27
Tokens per step	12.39	12.18	12.63
Task desc. tokens	10.02	10.09	10.04
Images	286.75	287.24	277.72
Actions	49.78	50.12	46.98
Images per action	6.08	6.02	6.12
Actions per step	7.6	7.72	7.72
Nav-interact ratio	9.19	9.25	8.13
Total objects	33.2	32.84	38.44
Mask coverage	0.17	0.17	0.15
Step-object coverage	0.86	0.85	0.88

Table 1: Average values of various quantitative aspects of ALFRED by split. See appendix for definitions.

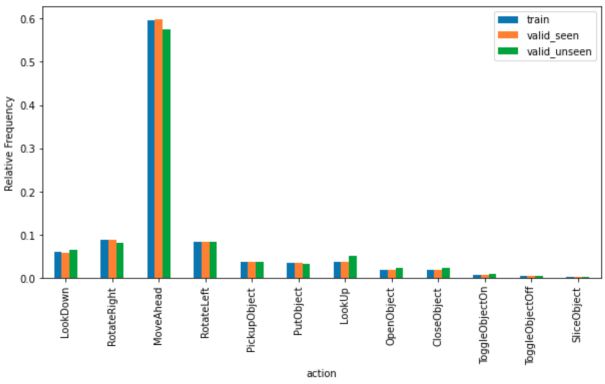

Figure 1: Relative frequency of each action type by split

Furthermore, Figure 1 shows the frequency of the 12 different actions (5 navigation actions + 7 interaction actions). Note that all splits have fairly equal frequency for all actions. Also, note that 60% of actions are "move ahead" actions which signifies the importance of a navigation module we may need in our model.

3.2.2 Textual Analysis

For textual analysis, we first identified out of vocabulary (OOV) words in the training and validation set using a vocabulary of 685k words defined by

spaCy. We found that less than 0.002% of all words were OOV in any split, indicating the dataset is already quite clean. Most of the OOV words were just misspelled (eg: "stovve"), indicating that it would be important to preprocess text using a simple spell checker before using it for downstream tasks.

We also found the top few synonyms used to describe objects in the dataset. This was performed by comparing the similarity of word vectors of all objects in the dataset with all common nouns identified in all task descriptions. The complete results are in the Jupyter notebook, but a few results are shown in Table 2. This reveals that our model will need to be robust enough to recognize synonyms of different words in order to be successful.

Furthermore, we analyzed how many objects present in the scene are directly referred to in the step-by-step instructions. The results are plotted in a histogram in Figure 2, which reveals that much less than 40% of objects in the scene are actually referred to in the instructions.

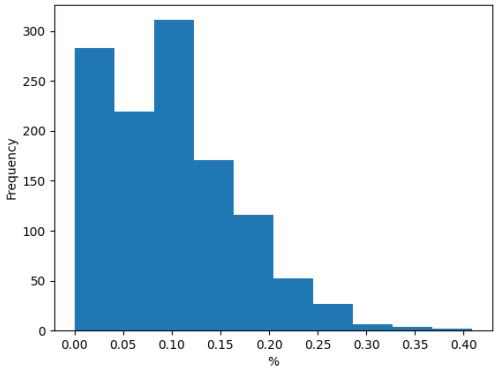

Figure 2: Percentage of objects referred to in instructions compared to objects in the scene

3.2.3 Visual Analysis

Due to compute constraints, most visual analysis was performed qualitatively by inspecting visual quality of different types of images. Figure 3 shows some sample RGB, depth and instance segmentation images. All images retrieved during simulation are of size 300 x 300.

Object Name	Synonyms used in task descriptions
Coffee Machine	Espresso Machine, Beverage Machine
Chair	Couch Chair, Sofa Chair
CD	DVD
Side Table	Corner Table
Butter Knife	Bread knife
Ottomon	Loveseat, Recliner
Fridge	Kitchen Fridge, Refrigerator
Poster	Wall Photo, Picture
Safe	Safety Box
Soap Bottle	Lotion Bottle

Table 2: Synonyms (i.e. closest words in embedding space) used in task descriptions of some objects in the dataset

3.2.4 Other Qualitative Analysis

In addition to the quantitative analysis, we also analyzed the solvability of the task. Our analysis in Figures 4 and 5 (in Appendix) shows that the unseen split of the validation set contains objects from the same classes as the training data, but could contain novel instances of those objects in novel environments. Since classes remain the same during training and testing times, the training data contains full information to solve the task, meaning a sufficiently intelligent agent should be able to solve the task, given training data.

3.3 Metrics

There are two primary metrics for evaluation. The first is "task success", which is a binary value indicating if the object positions and state changes correspond correctly to the goal-conditions of the task at the end of the action sequence. The second is "goal-condition success", which is the fraction of required goal-conditions that were completed at the end of the episode. Note that "task success" is true only if "goal-condition success" is 100%. Additionally, there exists a path-weighted version of these two metrics that considers the length of the episode, penalizing longer action sequences. For example, in the path-weighted version, an agent would receive half the score for taking twice as long as an expert to complete the task.

(a) (b) (c)

Figure 3: Sample RGB, depth, and instance segmentation images retrieved from AI2Thor simulator

4 Models (2 pages)

4.1 Baselines

4.1.1 Baselines and Other Methods

We plan to use the following baselines/other SOTA methods for comparison purposes. In the following, we only cite previous approaches (without explanation) as they have been described in detail in Section 2, but describe our proposed baseline approach in detail.

- **Seq-2-Seq (CNN-LSTM)** (Shridhar et al., 2020a)
- **Seq-2-Seq PM** (Shridhar et al., 2020a)
- **Modular Seq-2-Seq** (Corona et al., 2020)
- **MOCA** (Singh et al., 2020)
- **Seq-2-Seq RL (proposed):** Since we may likely use a reinforcement learning (RL) based approach, we plan to use the same Seq-2-Seq PM architecture and train it using a standard RL algorithm like PPO to get a baseline (Schulman et al., 2017). While it is hard to determine the exact reward function without any experimentation, some possible reward functions to try could be a combination of task completion, an intrinsic reward like curiosity (Burda et al., 2018), and possibly an imitation learning (IL) reward as proposed by Reddy et al. (2019). We could also add a reward that forces the agent to go towards objects in the order of the expert and interact with them in similar ways. Since the task is long and the reward could be sparse, we also intend to warm start the policy using IL as proposed in many previous approaches (Rajeswaran et al., 2017; Wang et al., 2019).

4.1.2 Proposed Ablations

Shridhar et al. (2020a) perform an input ablation study for their baseline Seq2Seq approach. They individually remove four inputs from the model: language, vision, the step-by-step instructions, and the goal statement, and evaluate the model's performance in each case. MOCA (Singh et al., 2020) performs the same ablation study as well, and we also plan to conduct this study on our model.

In addition to ablating the inputs to our model, we also plan to perform an ablation study over the training method. While our modeling plans have not been finalized, we currently are interested in exploring an RL-based approach. To empirically verify if RL performs better than IL, we plan to train two variants of our model: one trained using RL, and the other trained using simple IL like other approaches.

4.1.3 Metrics

We plan to use the following existing metrics to evaluate our approach.

1. **Task success (TS)**: TS is simply a binary value indicating whether the overall task has been completed. The task is considered complete if the final states and positions of objects of interest in the trajectory align with expected states.
2. **Goal-conditioned success (GCS)**: This is the ratio of goal-conditions completed at the end of the episode to the total number of goal-conditions required for task success. By definition, GCS is more granular than TS.
3. **Path Weighted Metrics**: We can also compute the path weighted versions of TS and GCS. So, if the model takes twice as many actions has the expert, the original TS and GCS scores would decrease by half.

In addition to computing the above metrics on the entire ALFRED task, we will also compute them at the sub-goal level by moving the agent through the expert trajectory before the sub-goal, and then letting it complete the sub-goal. This was also performed by Shridhar et al. (2020a).

Moreover, to better identify performance bottlenecks in ours and other approaches, we propose the following other metrics:

1. **Navigation Performance (NP)**: NP is the percentage of times the agent can come within a certain distance of the first object (and also face the correct object) it has to interact with for each sub goal. Note that we are only concerned with the first object of interaction because subsequent actions may involve interaction actions, making it difficult to reliably attribute credit of future success/failure to navigation alone. A high percentage would indicate that the agent is successfully able to at least navigate to the object of interest, indicating navigation may not necessarily be a performance bottleneck.

2. **Interaction Mask Prediction Performance (IMPP)**: Of all the times the agent predicts the correct interaction action near the correct object, IMPP is defined as the intersection over union score of the predicted instance segmentation mask with respect to the ground truth mask. Since IMPP is conditioned on the interaction action being correct, IMPP only measures the performance of the component that predicts the interaction mask. We also plan to measure IMPP grouped by object class to understand whether certain objects are more difficult to identify.

3. **Interaction Action Prediction Performance (IAPP)**: IAPP is the percent of times the model predicts the right interaction action when it has identified the instance segmentation mask of the correct object.

4. **Length-conditioned Success (LCS)**: LCS is the first n sub-goals successfully completed continuously. A low LCS indicates that the policy is struggling to complete tasks that have higher number of instructions.

Note that a recurring theme in most proposed metrics is to measure performance over each aspect needed to solve the task in as much isolation as possible. For example, we try to measure navigation and interaction performance separately by using NP, IMPP, and IAPP as proposed above.

In addition to the above base metrics, we also plan to use several of their more fine-grained variations to better understand model performance. Specifically, we plan to group all metric calculations by task type, environment type (i.e. room, kitchen, bathroom), expert trajectory length (i.e short, medium, high) to understand models at a more fine-grained level.

4.1.4 Empty results table

As mentioned in 4.1.1, we intend to compare our approach to five other approaches. Since Corona et al. (2020) do not report overall task success rate, we only use their goal-condition success metric for comparison. Note that the baseline approach is the highest-performing variant of their Seq2Seq model, specifically the model with both progress and sub-goal progress monitoring enabled. To the best of our knowledge, the methods we compare to are currently the only approaches that attempt the official ALFRED task as defined by Shridhar et al. (2020a) and are eligible for the ALFRED leaderboard. In particular, Storks et al. (2021) only evaluate on the **Stack & Place** task, and Saha et al. (2021) make use of depth map information from the simulator and use a different train/test split than the official benchmark, thus making both approaches not easily comparable. We report our overall results in Table 3. Additionally, we report results for our proposed intrinsic metrics on the validation set and compare to the baseline model in Table 4.

4.2 Proposed Approach

	Validation				Test			
	Seen		*Unseen*		*Seen*		*Unseen*	
Model	Task	Goal-Cond	Task	Goal-Cond	Task	Goal-Cond	Task	Goal-Cond
Baselines								
Seq2Seq (Shridhar et al., 2020a)	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	3.70 (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)
+ RL (proposed baseline)								
Modular (Corona et al., 2020)	-	-	-	-	-	8.80 (6.30)	-	7.20 (5.70)
MOCA (Singh et al., 2020)	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)
Ours								
Human	-	-	-	-	-	-	91.00 (85.80)	94.50 (87.60)
Ablations								
No Language								
No Vision								
Goal-Only								
Instructions-Only								
Trained using IL only								

Table 3: **Task and Goal-Condition Success Rate.** Corresponding path-weighted metrics are given in parentheses.

	Seen				*Unseen*			
Model	NP	IMPP	IAPP	LCS	NP	IMPP	IAPP	LCS
Baselines								
Seq2Seq								
+ PM Progress-Only								
+ PM SubGoal-Only								
+ PM Both								
+ RL (proposed baseline)								
Ours								
Ablations								
No Language								
No Vision								
Goal-Only								
Instructions-Only								
Trained using IL only								

Table 4: **Intrinsic Metrics - Validation.**

5 Results (1 page)

The columns above are just examples that should be expanded to include all metrics and baselines.

6 Analysis (2 pages)

This section should include at least two to three plots

6.1 Ablations and Their Implications

6.2 Qualitative Analysis and Examples

This section should likely contain a table of examples demonstrating how the current approach succeeds/fails.

References

Pieter Abbeel and Andrew Y Ng. 2010. Inverse reinforcement learning.

Ferran Alet, Tomás Lozano-Pérez, and Leslie P. Kaelbling. 2019. Modular meta-learning.

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443.

Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. 2018. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*.

Rodolfo Corona, Daniel Fried, Coline Devin, Dan Klein, and Trevor Darrell. 2020. Modularity improves out-of-domain instruction following. *arXiv preprint arXiv:2010.12764*.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, J. Moore, Matthew J. Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2018. Textworld: A learning environment for text-based games. In *CGW@IJCAI*.

Abhishek Das, Samyak Datta, Georgia Gkioxari, Stefan Lee, D. Parikh, and Dhruv Batra. 2018a. Embodied question answering. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2135–213509.

Abhishek Das, Georgia Gkioxari, Stefan Lee, D. Parikh, and Dhruv Batra. 2018b. Neural modular control for embodied question answering. *ArXiv*, abs/1810.11181.

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. *arXiv preprint arXiv:1806.02724*.

Michał Garmulewicz, Henryk Michalewski, and Piotr Miłoś. 2018. Expert-augmented actor-critic for vizdoom and montezumas revenge. *arXiv preprint arXiv:1809.03447*.

Weituo Hao, Chunyuan Li, Xiujun Li, Lawrence Carin, and Jianfeng Gao. 2020. Towards learning a generic agent for vision-and-language navigation via pre-training.

Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. 2018. Deep q-learning from demonstrations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Jonathan Ho and Stefano Ermon. 2016. Generative adversarial imitation learning. *arXiv preprint arXiv:1606.03476*.

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

Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. 2017. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35.

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.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Chih-Yao Ma, Jiasen Lu, Zuxuan Wu, Ghassan AlRegib, Zsolt Kira, Richard Socher, and Caiming Xiong. 2019. Self-monitoring navigation agent via auxiliary progress estimation.

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

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. 2015. Listen, attend, and walk: Neural mapping of navigational instructions to action sequences.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Ashvin Nair, Bob McGrew, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. 2018. Overcoming exploration in reinforcement learning with demonstrations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6292–6299. IEEE.

Duy-Kien Nguyen and Takayuki Okatani. 2018. Multi-task learning of hierarchical vision-language representation.

Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. 2017. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*.

Siddharth Reddy, Anca D Dragan, and Sergey Levine. 2019. Sqil: Imitation learning via reinforcement learning with sparse rewards. *arXiv preprint arXiv:1905.11108*.

Stéphane Ross and Drew Bagnell. 2010. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668. JMLR Workshop and Conference Proceedings.

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

Homagni Saha, Fateme Fotouhif, Qisai Liu, and Soumik Sarkar. 2021. A modular vision language navigation and manipulation framework for long horizon compositional tasks in indoor environment. *ArXiv*, abs/2101.07891.

Tim Salimans and Richard Chen. 2018. Learning montezuma's revenge from a single demonstration. *arXiv preprint arXiv:1812.03381*.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

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

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J. Hausknecht. 2020b. Alfworld: Aligning text and embodied environments for interactive learning. *ArXiv*, abs/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 G. Tür. 2021. Are we there yet? learning to localize in embodied instruction following. *ArXiv*, abs/2101.03431.

Weiyao Wang, Du Tran, and Matt Feiszli. 2020. What makes training multi-modal classification networks hard?

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

Xin Wang, Wenhan Xiong, Hongmin Wang, and William Yang Wang. 2018. Look before you leap: Bridging model-free and model-based reinforcement learning for planned-ahead vision-and-language navigation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 37–53.

Saim Wani, Shivansh Patel, Unnat Jain, Angel X Chang, and Manolis Savva. 2020. Multion: Benchmarking semantic map memory using multi-object navigation. *arXiv preprint arXiv:2012.03912*.

Chen Yu and Dana H. Ballard. 2004. On the integration of grounding language and learning objects.

Licheng Yu, Xinlei Chen, Georgia Gkioxari, Mohit Bansal, T. Berg, and Dhruv Batra. 2019. Multi-target embodied question answering. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6302–6311.

7 Appendix

7.1 Data Analysis Plots

Description of fields in Table 1:

1. Steps per directive: number of steps in each language directive
2. Tokens per step: number of words in each directive step
3. Task description tokens: number of words in directive task description
4. Images: number of images per demonstration
5. Actions: number of actions per demonstration
6. Images per action: number of images divided by number of actions per demonstration
7. Actions per step: number of actions divided by number of directive steps per demonstration
8. Nav-interact ratio: number of navigation actions divided by number of interaction actions per demonstration
9. Total objects: number of total objects in a scene per demonstration
10. Mask coverage: proportion of the image that is covered by the interaction mask per demonstration
11. Step-object coverage: proportion of interaction actions whose object of interest is mentioned in the step-by-step instructions, averaged over all interaction actions and language directives in the demonstration

In Figure 6, we see that ALFRED contains a roughly equal number of demonstrations for each type of task, and for the most part, a roughly equal proportion for each split. The validation data, especially the unseen portion, does have relatively less "Pick Two & Place" tasks than the training data. Additionally, the unseen portion has a significantly higher proportion of "Examine in Light" tasks than the other splits.

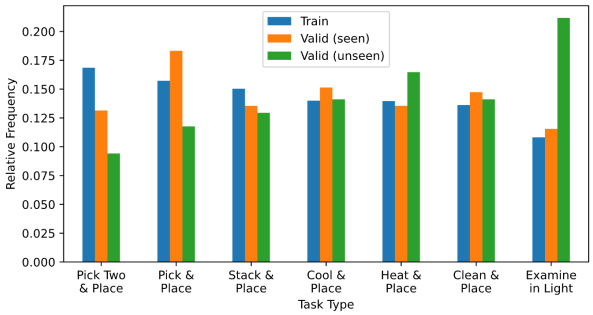

Figure 6: Relative frequency of each task type by split

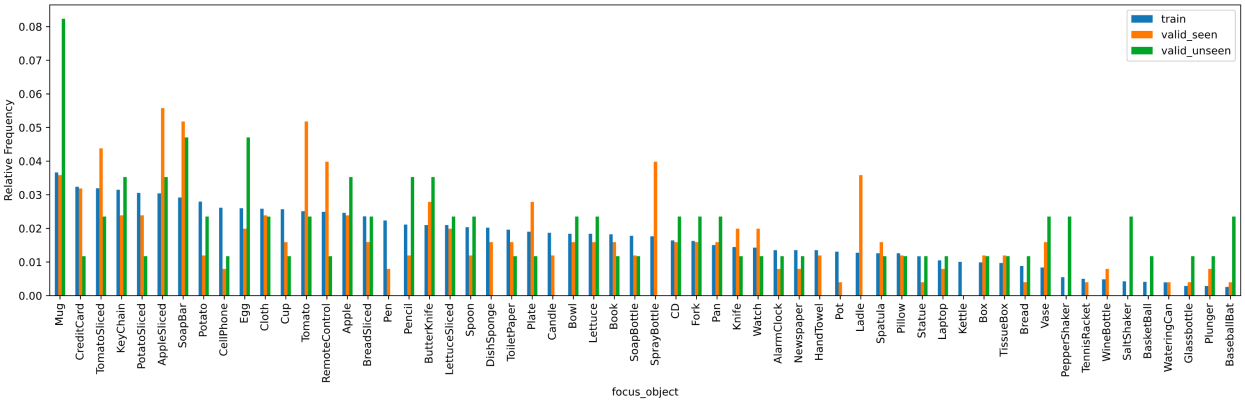

Figure 4: Relative frequency of focus objects used in different splits

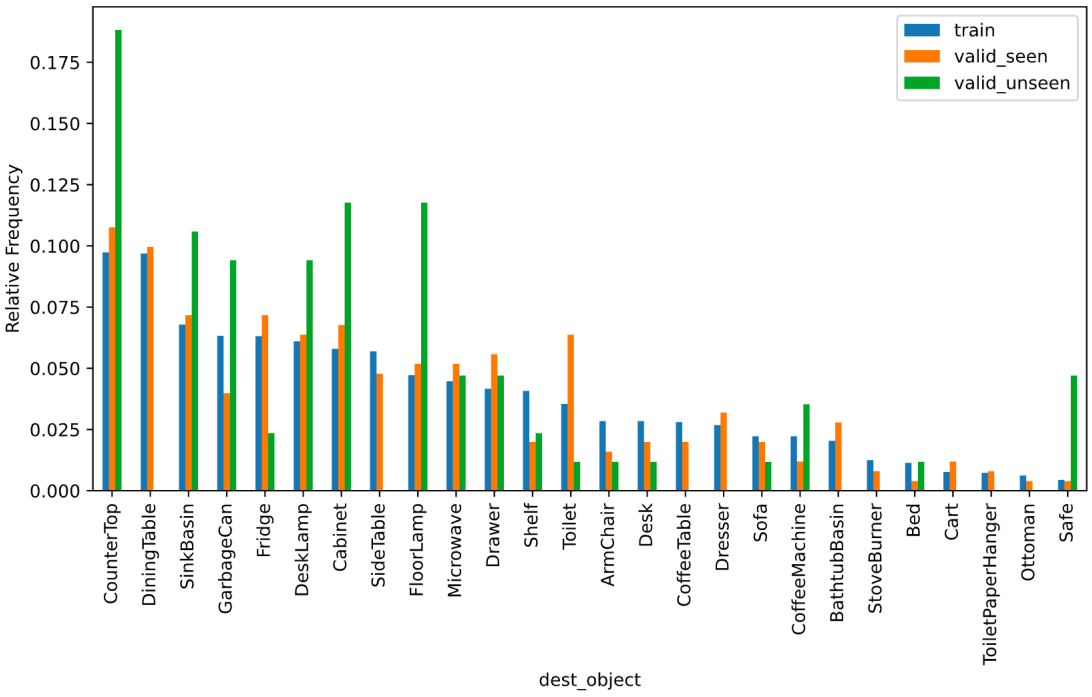

Figure 5: Relative frequency of destination objects used in different splits