Multimodal Contextualized Plan Prediction for Embodied Task Completion

Mert İnan[1], **Aishwarya Padmakumar**[2], **Spandana Gella**[2],
Patrick Lange[2], **Dilek Hakkani-Tur**[2]
[1] Computer Science Department, School of Computing and Information,
University of Pittsburgh, Pittsburgh, USA
[2] Amazon Alexa AI
mert.inan@pitt.edu, padmakua@amazon.com,
sgella@amazon.com, patlange@amazon.com, hakkanit@amazon.com

Abstract

Task planning is an important component of traditional robotics systems enabling robots to compose fine grained skills to perform more complex tasks. Recent work building systems for translating natural language to executable actions for task completion in simulated embodied agents is focused on directly predicting low level action sequences that would be expected to be directly executable by a physical robot. In this work, we instead focus on predicting a higher level plan representation for one such embodied task completion dataset - TEACh, under the assumption that techniques for high-level plan prediction from natural language are expected to be more transferable to physical robot systems . We demonstrate that better plans can be predicted using multimodal context, and that plan prediction and plan execution modules are likely dependent on each other and hence it may not be ideal to fully decouple them. Further, we benchmark execution of oracle plans to quantify the scope for improvement in plan prediction models.

1 Introduction

Task planning is an important component of traditional robotics systems enabling robots to compose fine grained skills to more complex tasks (Chen et al., 2010; Lemaignan et al., 2017; Jiang et al., 2019). Such robots typically have "skills" that abstract out a sequence of low-level motor control actions, for example, navigating from one point to another, picking up an object (Khandelwal et al., 2017). Skills are typically parameterized, for example, a navigation skill is likely to have parameters representing the source and destination. A task planning module typically has some notion of the state of the environment in which the robot is operating in and notions of the expected state changes to that environment when a skill is performed. Then, given a goal whose completion requires the chaining of skills, the task planner can reason about a sequence of skills that if executed can complete the task (Lipovetzky, 2014). Typically when such a sequence is executed, there will be additional modules that detect whether the individual skills were executed successfully and re-plan if failures are detected (Karapinar et al., 2012). We imagine our planning models presented in this paper as potential alternatives to classical planning in a future robotics system. In our case, we assume that a robot will have skills corresponding to various object interaction actions, which will additionally include navigating to the relevant object. While some of the skills used in this work, such as *slicing*, are not within the realm of current physical robots, we believe this is a useful level of abstraction at which a robot can learn to reason about task planning based on direct visual observations.

There are limitations to exploring new modelling techniques in physical robots as physical robots tend to be slow and are more sensitive to damage caused by poor choices of actions (Savva et al., 2019). As a result, simulated environments (Savva et al., 2019; Kolve et al., 2017; Chang et al., 2017) have become increasingly popular in recent years with a focus on training deep learning models from egocentric visual observations, and in some cases natural language instructions, to predict abstracted action sequences.

In this work, we focus on the TEACh benchmark (Padmakumar et al., 2022). This dataset consists of wizard-of-oz interactions between a user and an embodied agent collaborating via a natural language chat interface to complete household tasks. We focus on the Execution from Dialog History (EDH) task proposed for this dataset - given a segment of dialogue from the dataset as well as past actions and image observations, the model must predict subsequent actions in the environment to make progress on the relevant household task. The household tasks themselves are defined in terms of expected state changes in the environment and

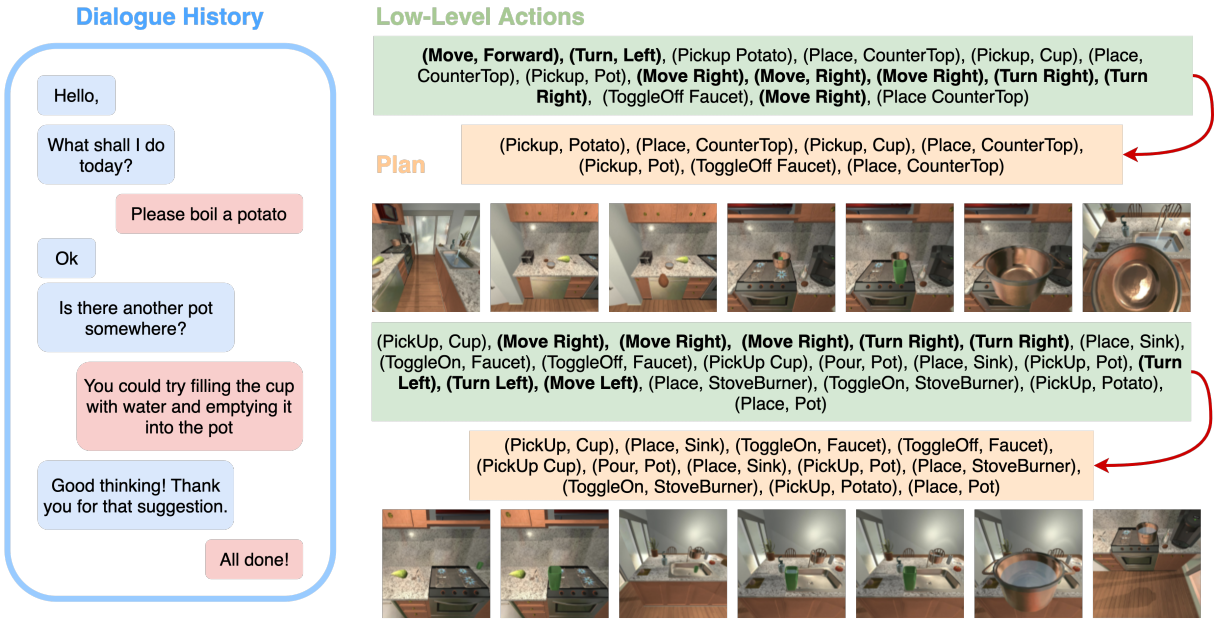

Figure 1: This figure depicts the main components of an EDH instance in the TEACh dataset. The dialogue history between the *Commander* (red) and the *Follower* (blue) happens concurrently with the low-level actions or the plan. The images are the history images of the driver robot in the simulator environment. For plan execution, all the low-level actions such as navigation are removed, resulting in a plan with high-level object interaction actions only.

it is shown in the original TEACh paper that it is difficult to develop a fully rule-based system to successfully execute the tasks (Padmakumar et al., 2022).

Prior work on other embodied task completion benchmarks have suggested that modular deep learning models with components for task planning, and semantic mapping and navigation, can outperform monolithic models that directly attempt to predict executable low-level actions based on language input and visual observations (Min et al., 2021; Jia et al., 2022). However, these results have been obtained on the ALFRED benchmark (Shridhar et al., 2020) where task planning can be formulated as a simple classification problem. In contrast, the TEACh dataset consists of more diverse tasks, including hierarchical and parameterized tasks and more diverse initial states, making planning non-trivial, as evidenced by the poor success rate of rule-based systems (Padmakumar et al., 2022).

In this work, we adapt the Episodic Transformer (Pashevich et al., 2021) (`E.T.`) model, originally proposed for predicting low level actions from multimodal observations for the ALFRED dataset to predict plans for the TEACh EDH task. We show that in combination with a rule based plan execution module, this model produces plans that are significantly more successful than plans generated purely based on language information. We further explore modifications to the basic `E.T.` architecture that remedy problems observed with plans predicted by the `E.T.` model. We compare this performance to oracle plans obtained from human demonstrations in the original dataset, explore some limitations of our plan representation, and further analyze the execution of oracle plans to identify causes of failure in rule based execution of plans. We believe this analysis will provide insight that will better guide the development of benchmarks and models for embodied task completion.

2 Related Work

Task planning has long been a standard component of physical robot architectures (Chen et al., 2010), particularly with general purpose service robots (Khandelwal et al., 2017; Peshkin et al., 2001). Classical task planners include a symbolic representation of the state of the world, a goal and skills the robot is capable of executing, and are expected to find a sequence of skills that when executed will transform the world into the goal state, typically using heuristic search algorithms (Lipovetzky, 2014). Over the years, research in planning has improved the symbolic representations used in planners (Fikes and Nilsson, 1971; McDermott, 1996; Gelfond and Kahl, 2014; Konidaris et al., 2018; Gopalan et al., 2020),

search algorithms (Hart et al., 1968; Helmert, 2004; Richter and Westphal, 2010) and handling uncertainty via probabilistic methods (Toussaint and Goerick, 2007; Bagchi et al., 1996; Ponzoni Carvalho Chanel et al., 2019). More recent work has focused on expanding beyond fully defined world representations by expanding to use common sense (Al-Moadhen et al., 2013) and open worlds (Jiang et al., 2019).

A parallel track in recent years is work in simulated environments that focus on training deep learning models directly from egocentric visual observations instead of symbolic representations. Some simulators include Habitat (Savva et al., 2019; Szot et al., 2021), AI2-THOR (Kolve et al., 2017), Matterport-3D (Chang et al., 2017), VirtualHome (Puig et al., 2018; Liao et al., 2019; Puig et al., 2020) and Chalet (Yan et al., 2018). Some tasks in such simulators are focused on learning sequences of discrete navigation actions including point goal navigation (Anderson et al., 2018a), object goal navigation (Batra et al., 2020), or motor control for visual object manipulation (Ehsani et al., 2021). Others involve both task planning and at least partial prediction of discrete navigation actions, for example transportation (Gan et al., 2020) and rearrangement (Kant et al., 2022). More related to our work, vision and language navigation learning to convert natural language route instructions into sequences of discrete navigation actions (Anderson et al., 2018b; Chen et al., 2019; Thomason et al., 2020) and embodied task completion involves predicting a sequence of both discrete navigation and object manipulation actions based on natural language instructions (Shridhar et al., 2020; Padmakumar et al., 2022; Suhr et al., 2019; Kim et al., 2020; Narayan-Chen et al., 2019). In this work, we adopt a modified version of the Execution from Dialog History (EDH) task defined in (Padmakumar et al., 2022) and focus on predicting only the necessary object interaction actions based on language input and egocentric visual observations, ignoring navigation actions.

There is some prior work focused on task planning in simulated environments for robots. Logeswaran et al. (2022) propose a model for task planning on the ALFRED dataset using GPT-2 which uses only the language input. Prior work has also explored task planning using only the language input in TEACh (Gella et al., 2022). We use the BART model from this work as a baseline in our work. In contrast to Gella et al. (2022), we explore the performance of multimodal planning models and evaluate based on success rate when combined with plan execution modules instead of relying on surface level metrics comparing ground truth and predicted plans. Some end-to-end models for the ALFRED dataset also have task planning modules as a component of their architecture (Min et al., 2021; Jia et al., 2022). However, due to the way the ALFRED dataset was created, task planning in ALFRED can be cast as a simple 7-way classification problem, whereas task plans for TEACh are more complex and often are very dependent on the initial state of the environment.

3 Task Setup

The TEACh dataset (Padmakumar et al., 2022) is a situated dialogue corpus, where two interlocutors communicate over a chatting interface simulating the interaction between a user (*Commander*) and embodied agent (*Follower*) to complete household tasks. The *Commander* has access to the task steps and the location information of objects, while only the *Follower* can interact with objects in the environment. The *Follower* has an egocentric view of the environment and can pose questions to the *Commander* about the location and task information. The *Follower* and must use this rich lexical and visual context to complete household tasks that require chaining long sequences of actions and reasoning about physical state changes.

In this paper, we focus on the Execution from Dialog History (EDH) benchmark on the TEACh dataset. Given past dialogue and actions from a human-human gameplay session, a model must predict subsequent actions the *Follower* would take in the environment until the next dialogue action. The model is evaluated by comparing the object state changes caused by the predicted action sequence with state changes caused by the ground truth action sequence. The action sequence predicted by the model is expected to be directly executable in the TEACh simulator. This action space includes discrete actions for navigation - `Move Forward`, `Turn Right`, `Turn Left`, `Move Backward`, `Strafe Left`, `Strafe Right`, `Look Up` and `Look Down` - as well as object interaction actions - `Pickup`, `Place`, `Open`, `Close`, `Toggle On`, `Toggle Off`, `Slice`, `Pour`. For object interaction actions, models must predict a relative (x, y) coordinate on the last ego-

centric image observation that are resolved by the underlying simulator into the object the action is to be executed upon. Using this, current neural models have at most a 10% success rate on the TEACh EDH task (Padmakumar et al., 2022) .

In contrast, in this paper, we modify the expected prediction from a model to be a "plan", which we define as being a sequence of object interaction actions paired with the object category of the object they are to be executed upon. An example for the task of boiling a potato is included in Figure 1. We can see that the original low level action sequence includes navigation actions for completing many object interaction actions. In order to pick up a potato, the agent must navigate near it and potentially alter where it is looking so that the potato is in view. However, when predicting a plan, a model can simply predict that the agent needs to pick up a potato and then a separate plan execution module (heuristic in our case but could be learned) positions the agents in the correct location to perform the action. While it is possible to define plans at an even higher level, say having subgoals such as filling a cup with water which then need to be broken down, we choose the level of object interaction actions as our chosen level of abstraction as plans of this level can be automatically created from ground truth actions sequences eliminating the need for annotation of plans.

Since the EDH task involves continuing a partially completed session, a particular EDH instance may show from the dialog that for example the cup is already filled with water. In this case, the plan the model would have to predict for this instance may only involve the last steps of pouring the water into the pot, setting it on the stove and adding the potato.

With many TEACh tasks, the task may also be parameterized, in which case plan prediction additionally involves identifying task parameters based on the dialog. For example, in the task of making breakfast, parameters are used to determine which dishes are to be prepared and how many of each are required. Changes in parameters change the plan the model is expected to predict, making the plan prediction problem less trivial than in earlier embodied task completion benchmarks. Further, the TEACh dataset includes a more diverse and difficult set of initial states, requiring reasoning to produce object interaction sequences to say open a cabinet to take out a knife from inside it or clearing out the sink to place a large plate.

During inference, at each time step, our plan prediction model is expected to predict one object interaction action and the corresponding object category for the object on which it is to be executed. This plan step is then executed by one of two possible plan execution modules described in section 5. Execution terminates either when a model predicts a special `Stop` action or if the model predicts 30 plan steps that result in an execution failure from the simulator, or reaches a limit of 100 plan steps. A plan step may fail execution for a variety of reasons. It may be infeasible, for example, trying to pick up a cabinet, a prerequisite step may not be completed (for example the `Slice` action is only feasible if the agent is holding a knife, or the execution module may not be able to find a feasible position for the agent from which to execute the action, for example the agent may not be positioned correctly in order to see and hence act on an object inside a drawer.

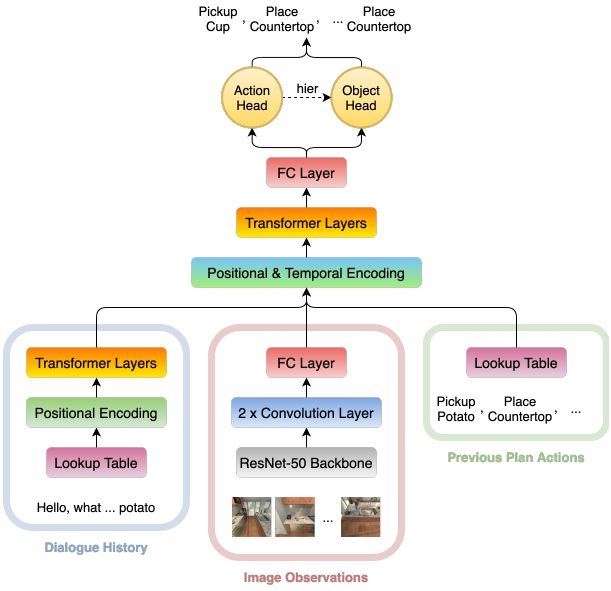

Figure 2: This is a depiction of the architecture for the `E.T.`-based models. The basic E.T. model does not have a connection between the action and object heads, but `E.T. Hierarchical`does. There are three main input processing components (dialogue history, image observation and previous plan actions). These are then input into the positional and temporal encodings and transformer layers to come up with the next feasible set of high level plan actions.

4 Plan Prediction Models

We modify the Episodic Transformer (`E.T.`) model (Pashevich et al., 2021), which has previ-

ously been used for action prediction in ALFRED as well as TEACh (Padmakumar et al., 2022), for plan prediction. The E.T. model consists of a transformer encoder for language input (which in our case is the EDH dialog history) and image observations encoded using a ResNet-50 backbone which are concatenated and passed through multimodal transformer layers and then converted via linear layers into action and object category predictions. The model is designed to predict as many actions and objects as there are image observations. At train time, the model receives image observations for the entire trajectory and is expected to produce actions for the entire trajectory. At inference time, the model receives image observations for the actions completed so far in the trajectory, as well as past actions as input, and the last action in its predicted sequence is used as the predicted action for the next time step. To modify the model for plan prediction instead of low level action prediction, we modify the training data retaining only the object interaction objects and image observations from the corresponding time steps. During inference, after each plan step is executed, the last image observation from this execution is appended to the visual input to obtain the next plan step.

We explore the following variants of the `E.T.` model:

- **`E.T.`**: Basic `E.T.` model described above.
- **`E.T. Hierarchical`**: The `E.T.` model has independent classification heads predicting the action and object for each time step - occasionally resulting in infeasible actions such as picking up a cabinet. Instead of having a local classifier per level, we explore sharing information between the two heads by concatenating the output of the action classifier head with the input of the object classifier head. This is hypothesized to be giving more information to the higher level object classifier to select the action based on action-object pair validity. The model is hypothesized to learn valid object-action pair patterns from valid samples, yet, this is not fully enforced wit a hard constraint on the classifier. This may still lead to invalid object-action pairings.
- **`E.T. + Mask`**: As an alternative to hierarchical learning, during inference, we explore whether the predicted action and object form an executable plan step. If not, we replace the action with the next most probable action that can be executed on the predicted object [1].

5 Plan Execution

In this work we focus on modeling plan prediction. While predicted plans can be compared to ground truth plans, the best measure of whether a plan is correct is whether executing it completes the task as expected. To evaluate our predicted plans in this manner, we pair them with two possible rule based plan execution modules. We plan to explore the task of developing machine learning models for plan execution in future work.

5.1 Direct Plan Execution

Given a plan step consisting of an action and associated object type, we need to identify the particular object the agent must manipulate. To do this, we use the heuristic of selecting the object of the desired type closest to the agent. We then use the navigation graph from the simulator to compute the shortest path to the object, navigate to it and attempt the predicted action.

5.2 Assisted Execution

Direct plan execution can fail for a variety of reasons. For instance, if the sink is full, and the predicted plan step requires placing a new item there, we need to empty it. While with our proposed level of abstraction we expect the plan prediction model to predict the necessary object interaction actions for this, we are also interested in seeing whether a model can be more successful if some of these details are abstracted out. During assisted execution, we identify common such situations that cause failures and execute additional actions to address the situation and increase the likelihood that the plan step can be successfully executed.

Specifically, we add the following assistive steps:

- For all actions, if the target object property change is already complete, do nothing to avoid an execution failure.
- *Pickup:* If the object is inside a receptacle (container), open the receptacle. After pickup, if a receptacle was opened, close it.

[1] We modify the action rather than the object as we expect object prediction to be easier as objects are visible in the image observations.

- *Place:* If the target receptacle is in a receptacle, take it out and place it on the counter first (for example, if we need to place something on a plate which is inside a drawer). If the target receptacle needs to be opened, open it and close after placement (for example, a drawer or microwave needs to be opened to place something inside). If a placement attempt fails, try removing existing contents of the receptacle one by one to make more space.

- *Open, Close:* Toggle off target object if relevant (for example, microwaves need to be turned off to open them).

- *ToggleOn, ToggleOff:* If the target is open, close it first (for example, microwaves need to be closed to turn them on).

- *Slice:* If the target is in a receptacle, first move it to the counter.

Additionally, we also attempt position adjustments to increase the chance of success.

6 Experiments

We evaluate our proposed plan prediction models on the EDH task of the TEACh dataset (Padmakumar et al., 2022). We experiment with each of the models in section 4 with each execution method in section 5. Additionally, we evaluate the following baseline and oracle conditions:

`Baseline`: Our baseline is a BART model, original proposed in (Gella et al., 2022), that uses only language input - the EDH dialogue history and predicts the entire plan as a sequence. As a result, it neither accounts for specific aspects of the environment that are not explicitly discussed, nor can it adapt to account for plan steps that failed to execute.

`Oracle`: As an upper bound to the success rate obtainable with each of our plan execution methods, we obtain oracle plans using the ground truth actions present in the EDH instance. We filter these actions sequences retaining only object interaction steps and converting object IDs to object types to match the plan representation used by our models.

Oracle with Object IDs (`CorefOracle`): To further test the extent to which our plan representation combined with our heuristic of selecting the closest object of a particular type limits performance, we also evaluate another oracle that we call `CorefOracle`. This produces plans containing object IDs instead of object types so that during plan execution, the execution module knows which object of a particular type must be manipulated. We do not compare to the TEACh EDH baselines in this paper as our execution methods access information that the TEACh baseline models are not allowed to access. However, all our models except the baseline have higher success rates than the TEACh baselines.

We use the following metrics for evaluation:

Edit distance: between ground truth and predicted plans, considering plan steps to have an edit distance of 1 if they match both in terms of the action and object and 0 otherwise [2].

Normalized Edit distance: Additionally, by definition, edit distance depends on the original length of the ground truth or predicted plans. As the length of the plan increases, the possibility of having a larger edit distance increases. This may be misleading in certain cases when the distribution of the lengths of the plans is skewed [3]. Hence, we explore two length-normalized edit distances to give more insight into the tuple differences.

- Ground-Truth-Length Normalized Edit Distance = $\frac{EditDistance}{LengthoftheGroundTruth}$

- Prediction-Length Normalized Edit Distance = $\frac{EditDistance}{LengthofthePredictedPlan}$

.

Fraction of valid plan steps: We consider a plan step to be valid if it is possible to execute the predicted action on an object of the predicted object type. For example `(Pickup, Potato)` would be valid while `(Pickup, Sink)` would not.

TEACh execution metrics:

- Success Rate (SR): Fraction of successful EDH instances, that is, EDH instances for which all desired object state changes occurred on executing the predicted plan.

- Goal Condition Success Rate (GC): Fraction of desired object state changes across EDH

[2] See appendix for an example.

[3] see appendix for detailed analysis of the skewed distribution.

Model	Execution	EDH Divided Val Split				EDH Divided Test Split			
		Seen		*Unseen*		*Seen*		*Unseen*	
		SR	GC	SR	GC	SR	GC	SR	GC
Baseline	Direct	11.26	13.67	7.51	11.03	7.19	9.62	8.87	9.54
	Assisted	11.92	17.27	8.91	12.19	9.80	12.30	10.27	12.31
E.T.	Direct	12.91	16.32	15.58	16.20	15.03	19.52	16.62	15.61
	Assisted	15.89	20.57	18.74	22.36	16.67	19.96	19.98	27.13
E.T. Hierarchical	Direct	14.24	15.67	16.23	17.27	14.71	17.97	17.27	20.30
	Assisted	18.21	20.45	18.09	24.53	17.97	23.67	19.70	25.82
E.T. + Mask	Direct	15.23	22.51	17.81	18.29	16.34	23.84	17.46	18.96
	Assisted	18.87	28.99	19.57	27.64	18.95	26.35	20.07	28.33
Oracle	Direct	61.92	63.64	55.57	58.48	54.58	53.58	56.77	58.01
	Assisted	68.87	72.13	61.97	63.07	61.44	62.49	63.21	64.87
CorefOracle	Direct	77.81	83.03	70.87	71.87	75.82	79.50	71.90	74.34
	Assisted	80.13	84.50	74.58	77.31	78.43	80.92	76.94	78.30

Table 1: Success rate (SR) and Goal Condition Success Rate (GC) of different models combined with different execution methods on the TEACh EDH task. Oracle performances are separated as upper bounds on the task. Best performance results are bolded for each metric and split in the specific execution method.

Model	Execution	Edit Distance	Frac Valid Plan Steps	GT-Norm ED	Pred-Norm ED
Baseline	Direct	4.87	94.61	1.39	1.22
	Assisted	4.87	94.61	1.39	1.22
E.T.	Direct	45.78	90.33	21.72	0.96
	Assisted	54.80	89.08	25.48	0.96
E.T. Hierarchical	Direct	45.08	88.25	21.49	0.96
	Assisted	51.52	86.84	23.75	0.95
E.T. + Mask	Direct	49.43	98.51	22.82	0.96
	Assisted	60.00	98.69	27.51	0.96

Table 2: Edit distance, fraction of valid plan steps, and Ground Truth (GT) and prediction (pred) length normalized edit distance (ED) of different models combined with different execution methods on the TEACh EDH `divided_valid_unseen` split. Oracles are not included here as by definition their plans are always valid and have edit distance 0.

instances that were completed through execution of predicted plans.

Since the TEACh test set is not public, we follow the standard protocol proposed in the TEACh codebase [4] of using a standardized division of the original validation sets into validation and test sets called the divided validation and divided test sets. Additionally, both for validation and testing, there are EDH instances situated in the same floorplans as training instances (seen), and those situated in completely new floorplans (unseen). The additional challenge imposed by unseen environments are visual differences with respect to the training data.

[4] https://github.com/alexa/teach

7 Results

We present TEACh execution metrics obtained from different models paired with different plan execution conditions in table 1. This additionally includes results from the two oracle models. For a subset of these conditions we train and perform inference with 3 random seeds and perform 2 sided Welch t-tests. Allowing for Bonferroni corrections over 4 tests, we find that `E.T. + Mask`is trending to be significantly better than the baseline with $p = 0.0381$ on the `divided_val_seen` split and $p = 0.0164$ on the `divided_test_seen` split. We did not find any statistically significant different between the `E.T. Hierarchical`and

E.T. + Maskmodels [5].

We observe that even with oracle plans combined with assisted plan execution, the success rate is not 100%. The highest oracle success rate is achieved using object IDs (CorefOracle) in assisted plan execution with 80.13% for validation and 78.43% for test splits. Our regular Oracle model sharing the same plan representation as our proposed models has a success rate of 68.87% for validation and 61.44% for test sets. This gap suggests that learning to disambiguate which particular instance of an object needs to be manipulated in each plan step plays a substantial role in eventual success rate. Despite using heuristic plan execution, for both oracle conditions we observe an unexpected drop in scores from seen to unseen splits. Further, even with oracle plans, assisted execution plays an important role in increasing success suggesting that low level position and placement adjustments play an important role in overall task success. Additionally, for all the cases in oracle performance, goal condition success rate is slightly higher than the regular success rate suggesting that even when plan execution fails for some EDH instances, enough progress is made to complete at least some of the desired state changes.

When examining models, the baseline starts with a success rate of around 11%. There is a drop in success rate from seen to unseen environments and assisted execution is primarily beneficial in unseen environments. The results for the E.T., E.T. Hierarchical and E.T. + Mask improve over the baseline but have a long way to go to meet even Oracle which uses the same plan representation. We see that the best performing model is E.T. + Mask in most cases. We conjecture that this is due to the fact that we enforce a hard constraint on the invalid action-object tuples so that the model never is able to predict invalid tuples such as (Pickup, Sink). This gives a performance improvement compared to the baseline.

Further, while assisted execution improves success rate in most cases, the improvements for models are much smaller than the Oracle condition. We find that in some cases the trends between the various E.T. models in direct execution do not match those in assisted execution. For example, in the divided_test_seen split, E.T. outperforms E.T. Hierarchical on direct execution but the reverse trend is observed with assistance. This suggests that the plan execution mechanism may affect the quality of plan prediction, at least in models like ours where the observations from executing one plan step are used in model input when obtaining the next plan step.

In the appendix, we also examine a breakdown of the success rate by task. We find that the oracle conditions have a lower success rate in tasks that either involve more task steps or more complex placement actions, although there is still a considerable gap between model and oracle performance. For the E.T. models, we see consistent increases in task level success rate for most tasks from E.T. to E.T. Hierarchical and E.T. to E.T. + Mask but the trend between E.T. Hierarchical and E.T. + Mask is less consistent. The baseline outperforms vanilla E.T. on some tasks but almost never outperforms E.T. Hierarchical or E.T. + Mask. From qualitatively comparing the predictions from the and the various E.T. models, we can see that the multimodal input is primarily beneficial in understanding how much of the task specified in the dialog history has already been completed in the action history. Plans generated by the model sometimes simply repeat a few plan steps from the history and fail to proceed further. Additionally, the multimodal input is able to help when the dialog history does not lay out detailed instructions on how a task is to be accomplished. For example, in the Water Plant task, the agent usually needs to fill some smaller container like a cup with water from the sink and pour this into the part. The model generates correct plans if the dialog history explicitly describes this process but not if it just involves the *Commander* asking the *Follower* to water the plant. However, the E.T. model appears to get prompted by the objects in the scene and is able to solve the task. Such visual prompting can also have the opposite effect though where in some cases the E.T. model ignores the language input and performs unrelated manipulations on easily visible objects, or ignores small objects in favour of larger, easier to see objects.

We additionally compare the performance of models using edit distance and fraction of valid plan steps on the divided_val_unseen split in Table 5. In contrast to success rates we observe

[5] We did not perform statistical comparisons across all pairs of conditions as it is expensive and time consuming to run inference with enough random seeds to allow for Bonferroni corrections as the number of tests grows.

that the baseline has a much lower edit distance of around 5 compared to the E.T. models at around 45 for direct execution. On further inspection, we find that this is because the E.T. models do not learn to predict when to stop. As such the plan ends when they hit the limit of plan step execution failures, which typically extends much longer than the ground truth plans. Their edit distance goes up with assisted execution further contrary to success rate as better execution means that more plan steps can be executed before reaching the limit of plan execution failures. We additionally find that the E.T. model starts out predicting more invalid plan steps than the baseline but this is fully corrected in E.T. + Mask by design. One counterintuitive observation is that the E.T. Hierarchical model, which was also designed to reduce invalid plan steps, is found to have the opposite effect.

Finally we examine the causes of plan execution failures of oracle plans, particularly the CorefOracle. We find that navigation failures are rare, accounting for less than 1% of the failure cases. We find that placement is particularly challenging, particularly when an object needs to be placed on another object that likely already contains other objects. We find that placing objects on a stove always fails, and placing items on dressers and coffee tables fails more than 50% of the time they are attempted. Opening and closing of cabinets and microwaves are also found to be challenging with failures primarily occurring because of fine adjustments to the agent's position that are difficult to calibrate heuristically.

7.1 Conclusion

We develop a model for multi-modal plan prediction for the TEACh dataset using the Episodic Transformer architecture and evaluate end to end performance on the TEACh EDH task in conjunction with heuristic plan execution modules. Our E.T. plan prediction models outperform a language only baseline but are quite far from oracle performance. The main cause of failure of our models is failing to accurately predict when to stop. We find that programmatic assistance in plan execution improves EDH success rates by a large margin on oracle plans but only to a limited extent on model generated plans.. We also find that plan model improvements do not maintain the same trend with different plan execution modules suggesting a dynamic interaction between plan prediction and plan execution improvements. Our work suggests opportunities for future work in plan representation, prediction and execution - as we show that our heuristic of navigating to the closest object substantially reduces the success rate of executing oracle plans, there is a significant gap between model and human generated plans, and our heuristic plan execution module is still only successful at executing human plans about 80% of the time.

References

Ahmed Al-Moadhen, Renxi Qiu, Michael Packianather, Ze Ji, and Rossi Setchi. 2013. Integrating robot task planner with common-sense knowledge base to improve the efficiency of planning. *Procedia Computer Science*, 22:211–220. 17th International Conference in Knowledge Based and Intelligent Information and Engineering Systems - KES2013.

Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. 2018a. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*.

Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. 2018b. Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sugato Bagchi, Gautam Biswas, and Kazuhiko Kawamura. 1996. Interactive task planning under uncertainty and goal changes. *Robotics and Autonomous Systems*, 18(1-2):157–167.

Dhruv Batra, Aaron Gokaslan, Aniruddha Kembhavi, Oleksandr Maksymets, Roozbeh Mottaghi, Manolis Savva, Alexander Toshev, and Erik Wijmans. 2020. Objectnav revisited: On evaluation of embodied agents navigating to objects. *arXiv preprint arXiv:2006.13171*.

Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. 2017. Matterport3d: Learning from rgb-d data in indoor environments. *International Conference on 3D Vision (3DV)*.

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.

Xiaoping Chen, Jianmin Ji, Jiehui Jiang, Guoqiang Jin, Feng Wang, and Jiongkun Xie. 2010. Developing high-level cognitive functions for service robots. In *AAMAS*, volume 10, pages 989–996.

Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. 2021. Manipulathor: A framework for visual object manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4497–4506.

Richard E Fikes and Nils J Nilsson. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3-4):189–208.

Chuang Gan, Jeremy Schwartz, Seth Alter, Martin Schrimpf, James Traer, Julian De Freitas, Jonas Kubilius, Abhishek Bhandwaldar, Nick Haber, Megumi Sano, et al. 2020. Threedworld: A platform for interactive multi-modal physical simulation. *arXiv preprint arXiv:2007.04954*.

Michael Gelfond and Yulia Kahl. 2014. *Knowledge representation, reasoning, and the design of intelligent agents: The answer-set programming approach*. Cambridge University Press.

Spandana Gella, Aishwarya Padmakumar, Patrick Lange, and Dilek Hakkani-Tur. 2022. Dialog Acts for Task-Driven Embodied Agents. In *Proceedings of the 23nd Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDial)*, pages 111–123.

N Gopalan, E Rosen, GD Konidaris, and S Tellex. 2020. Simultaneously learning transferable symbols and language groundings from perceptual data for instruction following. *Robotics: Science and Systems XVI*.

Peter E Hart, Nils J Nilsson, and Bertram Raphael. 1968. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107.

Malte Helmert. 2004. A planning heuristic based on causal graph analysis. In *ICAPS*, volume 16, pages 161–170.

Zhiwei Jia, Kaixiang Lin, Yizhou Zhao, Qiaozi Gao, Govind Thattai, and Gaurav Sukhatme. 2022. Learning to act with affordance-aware multimodal neural slam. *arXiv preprint arXiv:2201.09862*.

Yuqian Jiang, Nick Walker, Justin Hart, and Peter Stone. 2019. Open-world reasoning for service robots. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, pages 725–733.

Yash Kant, Arun Ramachandran, Sriram Yenamandra, Igor Gilitschenski, Dhruv Batra, Andrew Szot, and Harsh Agrawal. 2022. Housekeep: Tidying virtual households using commonsense reasoning. *arXiv preprint arXiv:2205.10712*.

Sertac Karapinar, Dogan Altan, and Sanem Sariel-Talay. 2012. A robust planning framework for cognitive robots. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Piyush Khandelwal, Shiqi Zhang, Jivko Sinapov, Matteo Leonetti, Jesse Thomason, Fangkai Yang, Ilaria Gori, Maxwell Svetlik, Priyanka Khante, Vladimir Lifschitz, et al. 2017. Bwibots: A platform for bridging the gap between ai and human–robot interaction research. *The International Journal of Robotics Research*, 36(5-7):635–659.

Hyounghun Kim, Abhaysinh Zala, Graham Burri, Hao Tan, and Mohit Bansal. 2020. Arramon: A joint navigation-assembly instruction interpretation task in dynamic environments. In *Findings of EMNLP*.

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*.

George Konidaris, Leslie Pack Kaelbling, and Tomas Lozano-Perez. 2018. From skills to symbols: Learning symbolic representations for abstract high-level planning. *J. Artif. Int. Res.*, 61(1):215–289.

Séverin Lemaignan, Mathieu Warnier, E Akin Sisbot, Aurélie Clodic, and Rachid Alami. 2017. Artificial cognition for social human–robot interaction: An implementation. *Artificial Intelligence*, 247:45–69.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union.

Yuan-Hong Liao, Xavier Puig, Marko Boben, Antonio Torralba, and Sanja Fidler. 2019. Synthesizing environment-aware activities via activity sketches. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Nir Lipovetzky. 2014. *Structure and inference in classical planning*. Lulu. com.

Lajanugen Logeswaran, Yao Fu, Moontae Lee, and Honglak Lee. 2022. Few-shot subgoal planning with language models. *arXiv preprint arXiv:2205.14288*.

Drew V McDermott. 1996. A heuristic estimator for means-ends analysis in planning. In *AIPS*, volume 96, pages 142–149.

So Yeon Min, Devendra Singh Chaplot, Pradeep Kumar Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. 2021. Film: Following instructions in language with modular methods. In *International Conference on Learning Representations*.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019. Collaborative dialogue in minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415.

Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gokhan Tur, and Dilek Hakkani-Tur. 2022. Teach: Task-driven embodied agents that chat. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 2017–2025.

Alexander Pashevich, Cordelia Schmid, and Chen Sun. 2021. Episodic transformer for vision-and-language navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15942–15952.

Michael A Peshkin, J Edward Colgate, Wit Wannasuphoprasit, Carl A Moore, R Brent Gillespie, and Prasad Akella. 2001. Cobot architecture. *IEEE Transactions on Robotics and Automation*, 17(4):377–390.

Caroline Ponzoni Carvalho Chanel, Alexandre Albore, Jorrit T'hooft, Charles Lesire, and Florent Teichteil-Königsbuch. 2019. Ample: an anytime planning and execution framework for dynamic and uncertain problems in robotics. *Autonomous Robots*, 43(1):37–62.

Xavier Puig, Kevin Ra, Marko Boben, Jiaman Li, Tingwu Wang, Sanja Fidler, and Antonio Torralba. 2018. Virtualhome: Simulating household activities via programs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8494–8502.

Xavier Puig, Tianmin Shu, Shuang Li, Zilin Wang, Joshua B. Tenenbaum, Sanja Fidler, and Antonio Torralba. 2020. Watch-and-help: A challenge for social perception and human-ai collaboration.

Silvia Richter and Matthias Westphal. 2010. The lama planner: Guiding cost-based anytime planning with landmarks. *Journal of Artificial Intelligence Research*, 39:127–177.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. 2019. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9339–9347.

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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10740–10749.

Alane Suhr, Claudia Yan, Jacob Schluger, Stanley Yu, Hadi Khader, Marwa Mouallem, Iris Zhang, and Yoav Artzi. 2019. Executing instructions in situated collaborative interactions. *arXiv preprint arXiv:1910.03655*.

Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. 2021. Habitat 2.0: Training home assistants to rearrange their habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Jesse Thomason, Michael Murray, Maya Cakmak, and Luke Zettlemoyer. 2020. Vision-and-dialog navigation. In *Conference on Robot Learning*, pages 394–406. PMLR.

Marc Toussaint and Christian Goerick. 2007. Probabilistic inference for structured planning in robotics. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3068–3073. IEEE.

Claudia Yan, Dipendra Misra, Andrew Bennnett, Aaron Walsman, Yonatan Bisk, and Yoav Artzi. 2018. Chalet: Cornell house agent learning environment.

A Training Hyperparameters

For training the `E.T.`, `E.T. + Mask` and `E.T. Hierarchical` methods, we retained hyperparameters from the original TEACh paper (Padmakumar et al., 2022), except the batch size, without further hyperparameter tuning, and used the largest batch size that could fit in a single GPU of a p3.8xlarge AWS EC2 instance. We used the AdamW optimizer with 0.33 weight decay with a learning rate of 1e 4 for the first 10 epochs and 1e 5 for the last 10 epochs. We trained all models for 20 epochs with a batch size of 3, and report results from the final epoch. We used two transformer layers for the language encoder with 12 attention heads and an embedding size of 768, and 2 multimodal transformer layers with 12 attention heads. We replace sampling with rotation permutations of our training dataset per epoch, ensuring that every train example is seen exactly once in our dataset. For language decoder in the transformer we use a drop-out of 0.1, and for the encoder we use a dropout of 0.1. The different `E.T.` models required 4 hours for preprocessing (extracting image features using the ResNet-50 backbone), about 2 hours per model for training using 4 GPUs of a p3.8xlarge AWS EC2 instance. At inference time we could use a maximum of 3 GPUs for inference as one GPU was required by the simulator. When using 3 GPUs of a p3.8xlarge AWS EC2 instance, E.T. models took about 11 hours to complete inference jointly on the `divided_val_seen`

and `divided_test_seen` splits and about 35 hours to complete inference jointly on the `divided_val_unseen` and `divided_test_unseen` splits. The time difference is due to the size of the various splits.

For the baseline BART model, we retain hyperparameters from the model presented in (Gella et al., 2022). We take the pretrained BART-base model from the Huggingface library [6] and finetune for 20 epochs using a batch size of 2 per GPU. Training was done using gradient accumulation across 4 GPUs of an p3.8xlarge AWS EC2 instance. We use the AdamW optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.99$, $\epsilon = 1e08$ and weight decay of 0.01. We use a learning rate of 5e05 with a linear warmup over 500 steps. The BART model can be finetuned in under an hour using all 4 GPUs of a p3.8xlarge AWS EC2 instance. We first performed inference on the BART model and saved the predicted plans to file before separately executing them in the AI2-THOR simulator. This process can also be completed in under an hour. Executing stored plans either in the case of the BART model or the oracle conditions took about 2.5 hours using 3 GPUs of a p3.8xlarge AWS EC2 instance for the combined `divided_val_seen` and `divided_test_seen` splits and about 8 hours for the combined `divided_val_unseen` and `divided_test_unseen` splits.

B Edit Distance

One of the metrics we use to evaluate predicted plans is edit distance (Levenshtein et al., 1966) to the ground truth plan. For computing edit distance, we treat each `(action, object)` pair as a token and compute the minimum number of edits - insertions, deletions and swaps - needed to convert the predicted plan to the ground truth plan. For example, given a ground truth plan `(Pickup, Mug), (Place, CoffeeMachine), (ToggleOn, CoffeeMachine)`, the following are the edit distances for a few sample predicted plans:

- Predicted plan: `(Pickup, CounterTop), (Place, CoffeeMachine), (ToggleOn, CoffeeMachine)` - Edit distance is 1 since the first step needs a swap of the object (we do not use assign partial credit for predicting the right action).
- Predicted Plan: `(Pickup, Mug), (ToggleOn, CoffeeMachine)` - Edit distance is 1 since the step `(Place, CoffeeMachine)` needs to be inserted.
- Predicted Plan: `(Pickup, Mug), (Place, CoffeeMachine), (ToggleOn, CoffeeMachine), (ToggleOff, CoffeeMachine)` - Edit distance is 1 since the final strep `(ToggleOff, CoffeeMachine)` needs to be deleted.

We can also see that depending on how task success is defined, the edit distance of a plan may not correlate well with task success. For example, suppose the task of making coffee only involves checking if the mug eventually has coffee, while all the above three plans have an edit distance of 1, the third plan alone has a task success of 1 but the other two will not succeed as both those plans fail to relocate the mug to the coffee machine, in which case it will not get filled with coffee.

We can see from Figure 3 that our distribution of ground truth is skewed towards shorter lengths, and they correlate with the edit distance linearly. This means that edit distance may not be comparable or insightful directly between different models if the output is of different length. We observe such difference between models in Table 5, where the baseline has a low edit distance compared to other ET models. In order to make them comparable without relying on the length of the ground truth, we normalize them according to their prediction lengths. This results in comparable results that are on the same scale, and shows the performance in terms of a corrected edit distance.

Prediction Length normalization shows that the edit distance in all methods are very similar to each other. Most are different from the ground truth length.

C Analyzing success rate across tasks

We include a breakdown of the success rate of the different models paired with the different execution methods at a task level in table 3. By examining oracle performance across tasks, we can confirm that plan execution is more likely to fail for tasks involving more or difficult placement steps. Boiling is the task causing the most oracle failures as it necessitates placing either a bowl in a microwave or a pot on the stove, both of which are challenging

[6] https://huggingface.co/

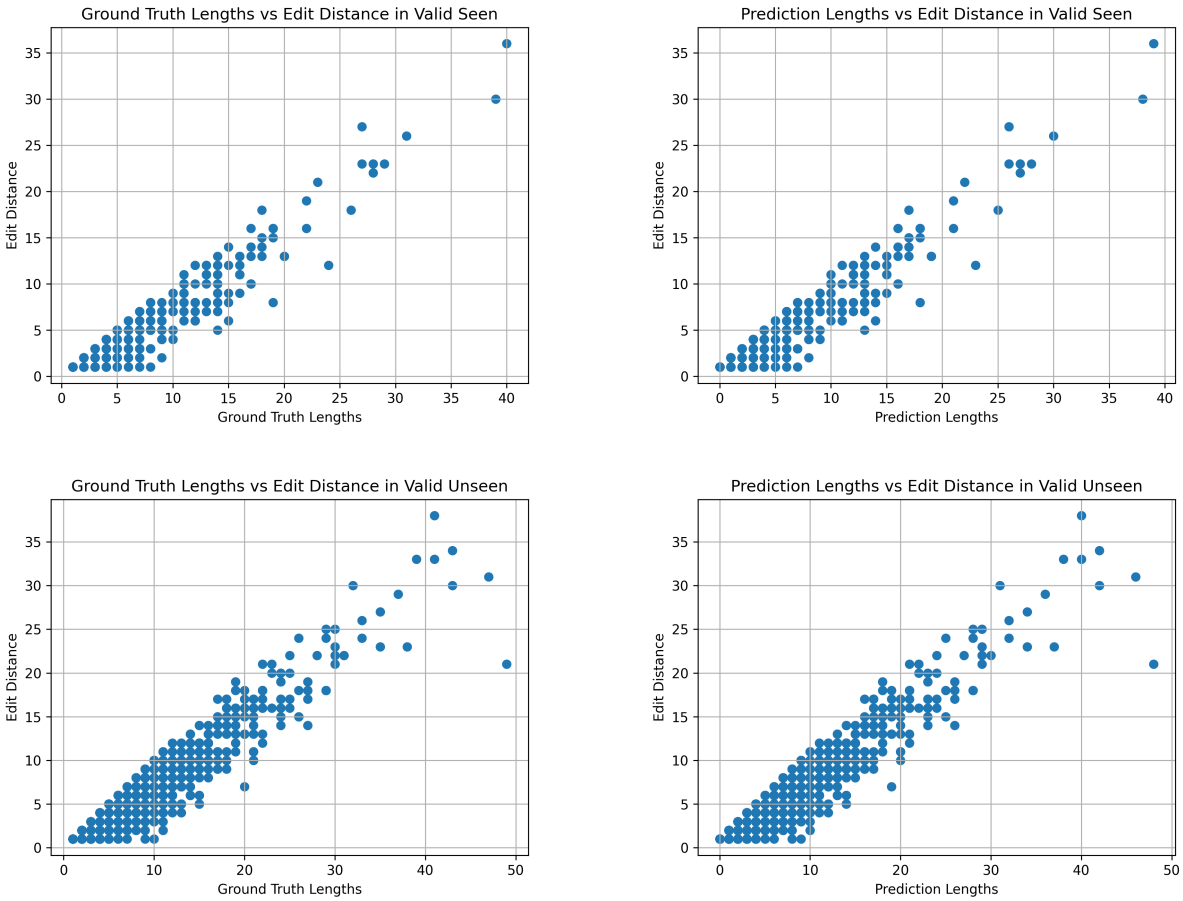

Figure 3: These plots show the distribution of edit distances and the ground truth and prediction lengths. As the length of the ground truth or predicted plan increases, the possibility of having a larger edit distance increases.

Task	Baseline		E.T.		E.T. Hier		E.T. Mask		Oracle		CorefOracle	
	Direct	Assisted	Direct	Assisted	Direct	Assisted	Direct	Assisted	Direct	Assisted	Direct	Assisted
Coffee	3.45	3.45	22.41	25.86	29.31	25.86	31.03	27.59	82.76	82.76	91.38	89.66
Water Plant	0.00	0.00	53.33	76.67	56.67	63.33	70.00	73.33	83.33	86.67	93.33	93.33
Plate Of Toast	11.11	11.11	6.67	10.00	10.00	15.56	6.67	13.33	50.00	64.44	75.56	81.11
Clean All X	3.23	3.23	22.58	20.97	24.19	20.97	29.03	24.19	69.35	75.81	72.58	77.42
Put All X On Y	15.56	17.78	11.11	20.00	17.78	15.56	15.56	24.44	48.89	55.56	66.67	84.44
Put All X In One Y	16.67	18.75	18.75	22.92	14.58	18.75	20.83	25.00	62.50	70.83	77.08	79.17
N Slices Of X In Y	7.95	11.36	15.91	22.73	10.23	15.91	14.77	17.05	60.23	65.91	75.00	78.41
N Cooked Slices Of X In Y	10.29	11.76	14.71	13.24	16.18	14.71	16.18	14.71	50.74	51.47	67.65	63.24
Boil X	12.50	15.00	20.00	20.00	15.00	12.50	15.00	22.50	40.00	40.00	57.50	57.50
Salad	5.44	8.16	12.24	14.97	13.61	14.29	12.93	12.93	44.22	52.38	63.95	64.63
Sandwich	9.29	10.71	10.71	10.71	7.86	13.57	11.43	15.00	51.43	62.14	71.43	80.00
Breakfast	2.58	3.09	15.46	20.10	17.53	20.10	18.56	20.10	57.22	62.89	65.98	73.20

Table 3: Breakdown of success rate by task for split `divided_val_unseen`. Some model names have been shortened due to space limitations.

in AI2-THOR due to physics constraints. Other tasks resulting in a higher rate of oracle failures are hierarchical tasks such as Salad, `Sandwich` and `Breakfast`.

For the E.T. models, we see consistent increases in task level success rate for most tasks from E.T. to E.T. Hierarchical and E.T. to E.T. + Mask but the trend between E.T. Hierarchical and E.T. + Mask is less consistent. The baseline outperforms vanilla E.T. on some tasks but almost never outperforms E.T. + Hierarchical or E.T. + Mask. We also notice many contrasts between tasks that the baseline is stronger on and tasks that the E.T. models are stronger on. For example in the `Water Plant` task, the baseline is never successful but the E.T. models perform much better on this than on other tasks.

We can also see that the benefits of execution assistance, both for models and for the oracles are for tasks involving multiple placement steps, despite the fact that placement is also the object interaction that has the highest failure rate even after assistance.

D Qualitative analysis

We include some qualitative examples in figures 4 and 5. We can see from the first example in figure 4 that the `E.T.` models are better able to identify how much of the task discussed in the dialog is already complete by making use of the visual context. The plan ends too early before the bread is sliced. In contrast, the `E.T.` plans start with the right action of slicing bread, but then as they do not learn when to stop, they start performing arbitrary actions on surrounding objects. This issue can lead to failure, as is seen in the first example in figure

Follower: what do i do
Commander: Hello, first, slice the bread
Commander: Bread and knife are in the fridge
Follower: knife??
Commander: In the fridge

History Actions: [('Pickup', 'Knife')]
Oracle plan: *[('Slice', 'Bread')]*

Baseline plan: [['Open', 'Fridge'], ['Pickup', 'Bread'], ['Place', 'CounterTop'], ['Pickup', 'Knife']]

E.T. plan: *[['Slice', 'Bread']*, ['Place', 'CounterTop'], ['Pickup', 'Bread'], ['Place', 'CounterTop'], ['Pickup', 'Bread'], ['Place', 'CounterTop'], ['Pickup', 'Bread'] …

E.T. + Hierarchical plan: *[['Slice', 'Bread']*, ['Place', 'Bread'], ['Pickup', 'Bread'], ['Place', 'CounterTop'], ['Pickup', 'Knife'], ['Place', 'CounterTop'], ['Pickup', 'Toaster'], …

E.T. + Mask plan: *[['Slice', 'Bread']*, ['Place', 'CounterTop'], ['Pickup', 'Bread'], ['Place', 'Toaster'], ['Pickup', 'Bread'], ['Place', 'Toaster'], …

Follower: what is my task today
Commander: Hello please water the plant.

History Actions: []
Oracle plan: *[('Pickup', 'Cup'), ('Pour', 'HousePlant')]*

Baseline plan: [['Pickup', 'Bowl'], ['Place', 'CounterTop'], ['Pickup', 'Kettle'], ['Place', 'Cup']]

E.T. plan: *[['Pickup', 'Bowl'], ['Pour', 'HousePlant']*, ['Place', 'CounterTop'], ['ToggleOn', 'Faucet'], ['ToggleOff', 'Faucet'], ['Pickup', 'Cup'], ['Pour', 'HousePlant'], ['Place', 'CounterTop'], ['ToggleOn', 'Faucet'], …

E.T. + Hierarchical plan: *[['Pickup', 'Bowl'], ['Pour', 'CounterTop']*, ['Place', 'CounterTop'], ['Pickup', 'Faucet'], ['Place', 'CounterTop'], ['Pickup', 'Faucet'], ['Place', 'Sink'], ['ToggleOn', 'Faucet'], ['ToggleOff', 'Faucet'], ['Pickup', 'Cup'], ['Pour', 'HousePlant'], ['Place', 'CounterTop'], ['Pickup', 'Faucet'],...

E.T. + Mask plan: *[['Pickup', 'Bowl'], ['Pour', 'HousePlant']*, ['Place', 'CounterTop'], ['ToggleOn', 'Faucet'], ['ToggleOff', 'Faucet'], ['Pickup', 'Cup'], ['Pour', 'HousePlant'], ['Place', 'CounterTop'], ['ToggleOn', 'Faucet'], …

Figure 4: Qualitative examples where the E.T. models outperform the baseline. The parts of the predicted plans that complete the necessary state changes are italicized. Multimodal context helps the model better identify how much of the task is already complete, and visual cues appear to help break down tasks requiring additional reasoning. However, the E.T. models also fail to stop after performing the necessary actions.

…
Follower: task
Commander: please place the newspapers on a single side table.
…

History Actions: []
Oracle plan: *[('Pickup', 'Newspaper'), ('Place', 'SideTable')]*

Baseline plan: *[['Pickup', 'Newspaper'], ['Place', 'SideTable']*, ['Pickup', 'RemoteControl']]

E.T. plan: *[['Pickup', 'Newspaper'], ['Place', 'SideTable']*, ['Pickup', 'Newspaper'], ['Place', 'CounterTop'], …*['Pickup', 'Newspaper']*]

E.T. Hierarchical plan: *[['Pickup', 'Newspaper'], ['Place', 'SideTable']*, ['Pickup', 'Newspaper'], … *['Pickup', 'Newspaper']*]

E.T. + Mask plan: *[['Pickup', 'Newspaper'], ['Place', 'SideTable']*, ['Pickup', 'Newspaper'], … *['Pickup', 'Newspaper']*]

Follower: What should I do today?
Commander: put pencils in safe
Commander: one on the desk

History Actions: []
Oracle plan: *[('Pickup', 'Pencil'), ('Open', 'Safe'), ('Place', 'Safe')]*

Baseline plan: *[['Pickup', 'Pencil'], ['Place', 'Safe']*]

E.T. plan: [['Pickup', 'Book'], ['Place', 'Bathtub'], ['Pickup', 'Pen'], ['Place', 'Desk'], ['Pickup', 'Candle'], …

E.T. + Hierarchical plan: [['Pickup', 'Book'], ['Place', 'Bathtub'], ['Pickup', 'RemoteControl'], ['Place', 'Floor'], ['Pickup', 'Book'], ['Place', 'Bathtub'], ['Pickup', 'Faucet'],...

E.T. + Mask plan: [['Pickup', 'Book'], ['Place', 'Bathtub'], ['Pickup', 'Pen'], ['Place', 'Desk'], ['Pickup', 'Candle'], …

Figure 5: Qualitative examples where the baseline outperforms the E.T. models. The parts of the predicted plans that complete the necessary state changes are italicized. These examples demonstrate cases where failing to stop can cause the environment to be in an incorrect state at the end of the episode, and visual cues can be distracting rather than helpful.

5 where the model reverses the correct actions it initially performed.

Visual cues also appear to help the `E.T.` models break down tasks that require more reasoning such as using a water in other utensils to water the plant, as is seen in the second example in figure 4. However, in some cases they can be distracting, as in the second example in figure 5 where the model is distracted by larger objects and fails to focus on pencils mentioned in the dialog, which the baseline correctly attends to.

E Complete Experimental results

The complete results with all metrics evaluated for all model and oracle conditions are included in tables 4, 5, 6, 7 for splits `divided_val_seen`, `divided_val_unseen`, `divided_test_seen` and `divided_test_unseen` respectively. When an oracle condition has a non-zero edit distance, this indicates that one or more oracle trajectories reached the execution failure limit, that is, the trajectory contained 30 plan steps which failed to get executed, at which point the rest of the trajectory is ignored.

Model	Execution	SR	GC	Edit Distance	Frac Valid Plan Steps	GT-Norm ED	Pred-Norm ED
Baseline	Direct	11.26	13.67	4.87	94.61	1.39	1.22
	Assisted	11.92	17.27	4.87	94.61	1.39	1.22
E.T.	Direct	12.91	16.32	45.78	90.33	21.72	0.96
	Assisted	15.89	20.57	54.80	89.08	25.48	0.96
E.T. Hierarchical	Direct	14.24	15.67	45.08	88.25	21.49	0.96
	Assisted	18.21	20.45	51.52	86.84	23.75	0.95
E.T. + Mask	Direct	15.23	22.51	49.43	98.51	22.82	0.96
	Assisted	18.87	28.99	60.00	98.69	27.51	0.96
Oracle	Direct	61.92	63.64	0.00	85.96	0.00	0.00
	Assisted	68.87	72.13	0.01	85.93	0.01	0.00
CorefOracle	Direct	77.81	83.03	0.01	86.13	0.00	0.00
	Assisted	80.13	84.50	0.03	86.08	0.01	0.02

Table 4: Complete results of different models combined with different execution methods on the TEACh EDH `divided_val_seen` split.

Model	Execution	SR	GC	Edit Distance	Frac Valid Plan Steps	GT-Norm ED	Pred-Norm ED
Baseline	Direct	7.51	11.03	5.78	95.76	1.56	1.42
	Assisted	8.91	12.19	5.78	95.80	1.56	1.43
E.T.	Direct	15.58	16.20	39.89	89.48	18.20	0.95
	Assisted	18.74	22.36	47.21	88.78	20.83	0.95
E.T. Hierarchical	Direct	16.23	17.27	41.91	88.93	19.24	0.94
	Assisted	18.09	24.53	49.37	88.78	22.26	0.95
E.T. + Mask	Direct	17.81	18.29	43.48	98.79	19.68	0.95
	Assisted	19.57	27.64	52.26	98.80	22.95	0.95
Oracle	Direct	55.57	58.48	0.00	85.21	0.00	0.00
	Assisted	61.97	63.07	0.01	85.18	0.00	0.01
CorefOracle	Direct	70.87	71.87	0.13	85.72	0.01	0.02
	Assisted	74.58	77.31	0.11	85.58	0.01	0.02

Table 5: Complete results of different models combined with different execution methods on the TEACh EDH `divided_val_unseen` split.

Model	Execution	SR	GC	Edit Distance	Frac Valid Plan Steps	GT-Norm ED	Pred-Norm ED
Baseline	Direct	7.19	9.62	5.75	95.55	1.58	1.47
	Assisted	9.80	12.30	5.75	95.55	1.58	1.47
E.T.	Direct	15.03	19.52	44.10	88.79	21.18	0.95
	Assisted	16.67	19.96	52.16	88.43	24.00	0.95
E.T. Hierarchical	Direct	14.71	17.97	43.39	89.82	19.68	0.94
	Assisted	17.97	23.67	55.06	88.18	24.97	0.96
E.T. + Mask	Direct	16.34	23.84	49.01	99.03	22.89	0.95
	Assisted	18.95	26.35	62.47	98.97	28.85	0.95
Oracle	Direct	54.58	53.58	0.01	87.54	0.00	0.00
	Assisted	61.44	62.49	0.00	87.55	0.00	0.00
CorefOracle	Direct	75.82	79.50	0.13	87.97	0.01	0.04
	Assisted	78.43	80.92	0.11	87.84	0.01	0.04

Table 6: Complete results of different models combined with different execution methods on the TEACh EDH `divided_test_seen` split.

Model	Execution	SR	GC	Edit Distance	Frac Valid Plan Steps	GT-Norm ED	Pred-Norm ED
Baseline	Direct	8.87	9.54	5.46	94.99	1.45	1.34
	Assisted	10.27	12.31	5.45	95.03	1.45	1.35
E.T.	Direct	16.62	15.61	41.37	89.88	18.74	0.95
	Assisted	19.98	27.13	47.36	89.30	20.71	0.95
E.T. Hierarchical	Direct	17.27	20.30	41.67	88.69	18.90	0.95
	Assisted	19.70	25.82	48.44	88.18	20.93	0.95
E.T. + Mask	Direct	17.46	18.96	44.80	99.02	20.27	0.95
	Assisted	20.07	28.33	53.80	98.98	23.29	0.96
Oracle	Direct	56.77	58.01	0.00	85.77	0.00	0.00
	Assisted	63.21	64.87	0.02	85.71	0.01	0.01
CorefOracle	Direct	71.90	74.34	0.14	86.17	0.01	0.02
	Assisted	76.94	78.30	0.13	86.10	0.01	0.02

Table 7: Complete results of different models combined with different execution methods on the TEACh EDH `divided_test_unseen` split.