Multimodal Analysis of Embodied Instruction Following on ALFRED

Justin Dannemiller* **Haoyang He*** **Conner Pulling*** **Eduardo Trevino*** **Renos Zabounidis***
{jdannemi, hhe2, cpulling, eatrevin, renosz}@andrew.cmu.edu

Abstract

Embodied Instruction Following (EIF) tasks focus on agents that navigate and interact with environments based on natural language instructions. Unlike other embodied intelligence tasks, EIF requires agents to adapt to unseen environments and interact dynamically, creating significant challenges. This paper explores the performance of EIF methods Prompter and FILM with updated foundational models for computer vision and natural language processing. Using the ALFRED dataset, a simulated environment for vision-and-language navigation, we evaluate the impact of integrating DepthAnything and MaskDINO. Our findings indicate that ground-truth depth and instance segmentation significantly boost performance, with DepthAnything outperforming Prompters depth estimator by 40% (improvement in average MSE) without fine-tuning, and MaskDino achieving competitive but slightly lower results compared to Prompter's existing MaskRcnn. These results demonstrate the potential of updated models to improve embodied agents' adaptability and effectiveness in complex tasks. With further finetuning, these methods could significantly surpass their predecessors.

1 Introduction and Problem Definition (1-1.25 pages)

Developing embodied agents is crucial for advancing artificial intelligence, particularly in enhancing interactions between AI and the physical world. These agents, including robots and virtual entities, are designed to operate in both real-world and simulated environments. This development is essential for creating systems that can understand and manipulate their surroundings, interact seamlessly with humans and other agents, and autonomously perform complex tasks. A specific area of interest is Embodied Instruction Following (EIF), where agents follow natural language instructions to execute long-horizon tasks that require both navigation and interaction with the environment. Unlike other tasks in embodied intelligence, EIF challenges agents to navigate unseen environments and interact dynamically with them, handling variations in tasks, language instructions, and actions. This complexity makes EIF one of the most human-like and challenging tasks, pushing the boundaries of adaptability, autonomy, and usefulness of AI systems in everyday applications.

Initial efforts in EIF predominantly employed end-to-end models via imitation learning or reinforcement learning (Pashevich et al., 2021). Due to the high training costs and low sample efficiency of these algorithms, along with their poor generalization outside of their training sets, state of the art methods have replaced this end to end approach with more modular approaches (Inoue and Ohashi, 2022; Blukis et al., 2022; Sarch et al., 2023; Jain et al., 2024; Kim et al., 2023; Song et al., 2023; Ding et al., 2023; Gu et al., 2023). These approaches enable the use techinques from computer vision and natural language processing to process raw RGB into meaningful features using object detection, scene recognition, semantic segmentation, and natural language parsing.

Recent advances in natural language processing and computer vision have increased the performance and generalizability of these techniques by leveraging vast amounts of unlabeled data using unsupervised losses. Existing EIF methods are agnostic to their modules, and thus their performance may be improved by integrating these foundational models. In this paper, we investigate how Prompter and FILM, two state-of-the-art EIF methods, improve when leveraging updated foundational models. We evaluate our agents on the ALFRED dataset (Shridhar et al., 2020), a collection of simulated environments which targets

*Everyone Contributed Equally – Alphabetical order

the intersection of vision-and-language navigation (VLN) and task-oriented interaction within these simulated domestic settings.

In this work, we evaluate how Prompter and FILM improve with an updated vision pipeline. We ablate over ground-truth depth and instance segmentation and find that ground-truth depth and instance segmentation drastically improve model performance. Furthermore, the gap between Prompter and FILM vanish when ground truth depth and segmentation are used. With this as motivation, we incorporate two state of the art models, DepthAnything (Yang et al., 2024) and MaskDino (Li et al., 2023a). Attempts to successfully fine-tune DepthAnything on ALFRED failed, and thus we evaluate the performance of DepthAnything zero-shot and find that it outperforms Prompter depth estimator by 40% while not being fine-tuned. Additionally, we fine-tune MaskDino (Li et al., 2023a) on a fraction of the training set (1/20th) and find that it has competitive but worse performance than the existing MaskRcnn model used by Prompter..

2 Related Work and Background

2.0.1 Foundation Models

GPT-4 (Achiam et al., 2023) is a group of state-of-the-art multimodal foundation models with exceptional reasoning capabilities in language and vision, but is not open source.

LLaMA-2 (Touvron et al., 2023) is a group of large language models that have state-of-the-art performance among open source LLMs.

SAM (Kirillov et al., 2023) is a foundation model for image segmentation in 2D, and SAM3D (Yang et al., 2023) extends SAM into the 3D space for 3D segmentation.

CogVLM (Wang et al., 2023) is a recent powerful open-source vision-language foundation model. CogAgent (Hong et al., 2023) is a closely-related vision-language model trained specifically for understanding of graphical user interfaces.

2.0.2 Multimodal Robotics with LLMs

EgoTV (Hazra et al., 2023) proposes a Neuro-Symbolic Grounding (NSG) approach that translates natural language into a graph of symbolic queries to capture the compositional and temporal structure of tasks.

ManipLLM (Li et al., 2023b) incorporates LLMs to reason with the object-centric manipulation task by encoding and projecting visual inputs into the language space and fine-tune a LLM with both the output template prompt and the projected vision as inputs.

SAGE (Geng et al., 2023) performs instruction-following manipulation tasks by incorporates visual inputs into natural language descriptions using VLMs, then combine it with the instruction to ask for planning from LLMs.

2.0.3 Instruction-Following Embodied Agents

FILM (Min et al., 2021) uses a modular method to build a structured representation of the environment that builds a semantic map of the scene and performs exploration with a semantic search policy.

Episodic Transformer (Pashevich et al., 2021) is an end-to-end multimodal transformer that encodes language inputs and the full episode history of visual observations and actions to output actions and objects.

BAS (Chiang et al., 2021) examines the alignment between vision and language modalities of existing methods on the ALFRED dataset by proposing an intrinsics metric boundary adherence score.

Prompter (Inoue and Ohashi, 2022) proposed an updated FILM++ which doubled FILM's performance, then proposed an update to FILM's framework by replacing its semantic search module with language model prompting.

HLSM (Blukis et al., 2022) proposes a persistent spatial semantic representation and uses hierarchical reasoning to effectively execute long term tasks by separating the reasoning into a learned high-level controller defining subgoals and low-level controllers, including engineered and learned controllers, executing the subgoals.

HELPER (Sarch et al., 2023) is a framework with an external memory of language-program pairs that parses dialogues into action programs through retrieval-augmented LLM prompting.

ODIN (Jain et al., 2024) challenges the belief that 2D and 3D perception requires unique architectures by introducing a model that can segment and label both 2D RGB images and 3D point clouds, using a transformer architecture that alternates between 2D within-view and 3D cross-view information fusion. It shows enhanced performance for the ALFRED benchmark when incorporated in the HELPER framework as replacement for 2D perception module, demonstrating its advantage in 3D understanding with 2D inputs.

CAPEAM (Kim et al., 2023) combines Context-Aware Planning separating instructions into subgoals and Environment-Aware Memory that caches the detected object locations in the semantic spatial map to enhance the agent performance with the knowledge of consequences of previous actions.

LLM-Planner (Song et al., 2023) consolidates the visual inputs and environment feedback of the embodied agent into the language modalities by textual descriptions, and repeatedly query LLMs to make action decisions.

ECL (Ding et al., 2023) introduces a framework enabling robotic agents to learn and apply visual concepts and depth understanding through instruction following, mimicking how humans learn from interaction and demonstration.

ConceptGraphs (Gu et al., 2023) presents a pipeline for grounding visual inputs in an open-vocabulary 3D scene graph, using VLMs and LLMs to caption each of the segmented objects and record them as node, using LLMs to build potential interaction reasonings between each nodes as edges, and then utilizing LLMs to reason across the built scene graphs to execute complex reasoning tasks, while using SLAM to track a mapping of its environment. It demonstrates great potentials for Instruction-Following Embodied Agents, but has the notable limitation of large inference costs.

2.0.4 Other Relevant Studies

CLIP (Radford et al., 2021) introduces a framework for alignment between texts and images using contrastive vision-language pre-training.

CLIP-Dissect (Oikarinen and Weng, 2022) automatically describe the function of individual hidden neurons inside vision networks leveraging the CLIP model to enhance interpretability without the need for labeled data.

Matryoshka Representation Learning (Kusupati et al., 2022) encodes information at different granularities that allows a single embedding to adapt to the computational requirements of various downstream tasks.

RelaTe (Bhagat et al., 2023) presents an approach to integrate a symbolic knowledge graph into a state-of-the-art recognition model, combining with a neuro-symbolic architecture and training approach that incorporates extra relationships from a few examples, that significantly enhances few-shot classification by leveraging interconnected entity presence.

Concept Policy Model (Zabounidis et al., 2023) incorporates interpretable concepts from a domain expert into models trained through multi-agent reinforcement learning by requiring the model to first predict such concepts then utilize them for decision making.

SplaTAM (Keetha et al., 2023) enables precise camera tracking and high-fidelity reconstruction for dense SLAM by online optimization of 3D Gaussian Splatting using differentiable rendering.

OpenIns3D (Huang et al., 2023) introduces a 3D open-vocabulary instance segmentation method requiring no 2D inputs, and maps the inputs to 2D synthetic images to incorporate existing 2D open-vocabulary instance segmentation methods.

Depth Anything (Yang et al., 2024) presents a robust solution for monocular depth estimation using large-scale unlabeled data. Unlike traditional approaches that rely on depth sensors or stereo matching, Depth Anything leverages a data engine to automatically annotate vast amounts of unlabeled images, significantly expanding data coverage and reducing generalization error. Its zero-shot capability and extensive evaluations across multiple datasets demonstrate its strong generalization and robustness in varied conditions.

Mask DINO (Li et al., 2023a) extends the DINO framework by incorporating a mask prediction branch to support all image segmentation tasks, including instance, panoptic, and semantic segmentation. It utilizes query embeddings from DINO to predict a set of binary masks through a dot-product with a high-resolution pixel embedding map, making it scalable and efficient. Mask DINO is noted for its simplicity and adaptability to joint large-scale detection and segmentation datasets, significantly outperforming existing specialized segmentation methods. Its notable achievements include setting new benchmarks on instance segmentation (54.5 AP on COCO), panoptic segmentation (59.4 PQ on COCO), and semantic segmentation (60.8 mIoU on ADE20K), making it the best-performing model among those with fewer than one billion parameters.

3 Task Setup and Data

3.1 Environment

We evaluate our work on the ALFRED set of environments (Shridhar et al., 2021). ALFRED combines natural language processing and vision-and-language navigation to assess an agent's ability to follow complex, sequential instructions using both

visual and linguistic cues. The dataset emphasizes navigating and interacting with objects in simulated indoor environments, requiring understanding of object properties, spatial relationships, and contextual information. ALFRED introduces challenges in long-horizon task planning and decision-making for achieving multi-step objectives, and it employs ProcTHOR, an extension of AI2-THOR, for procedural generation of diverse indoor settings, crucial for VLN and task-oriented research. This approach ensures a broad spectrum of environment configurations, countering overfitting and fostering model generalization across dynamic scenarios, essential for advancing embodied AI capabilities.

3.2 Collecting Training Data

We collect training data using the following methodology suggested by the FILM paper. Below is our methodology:

1. **Initialization**: Run the Prompter with complete ground truth and a random semantic policy. Execute the current action, and at each step, determine the next action with a probability of $\frac{1}{3}$.

2. **Randomized Rotation**: At each step, set the horizon to zero and rotate left four times, recording the number of objects seen from each orientation. Rotate towards the direction where the most objects are observed.

3. **Data Capture**: Save the RGB image, segmentation mask, and depth mask at horizons 0 and 15 degrees (where 0 represents the agent looking straight ahead). With a 50% probability, save these masks at additional horizons: 30, 45, and 60 degrees.

4. **Full Rotation Check**: If selected, rotate the agent 180 degrees and save the masks at horizons 30, 45, and 60 degrees. Return the agent to the original pose.

5. **Interaction Actions**: If the current action involves interaction, with a 50% probability, save the masks at horizons 0, 15, 30, 45, and 60 degrees. Rotate the agent 180 degrees again, and save the masks at the same horizons before returning to the original position.

6. **Repetitions**: Follow the above steps, repeating actions based on the Prompter's instructions, to ensure comprehensive data collection.

This methodology was used to collect data in 2,047 of the 20,000 training episodes. FILM uses the full 20,000; however, we found that on only 1,000 trajectories, we already collected over 300GB worth of rgb images, depth maps, and instance segementations.

3.3 Depth Eval. Setup

To evaluate the effectiveness of our depth estimation models, we constructed a dataset using valid unseen data from 128 episodes containing synchronized RGB images, baseline depth maps, and ground truth depth maps. This dataset was designed to test the models under diverse conditions to assess their generalization capabilities.

3.3.1 Dataset Preparation

Our dataset was prepared by selecting episodes that include a variety of scene types, ensuring a robust challenge for the models. Each episode consists of ground truth depth maps and corresponding RGB images used for depth prediction.

3.3.2 Evaluation Protocol

We employed the following procedure to evaluate the models:

1. RGB images were converted to grayscale to normalize the input data.

2. Depth predictions were made from the RGB images using both the "Depth Anything" model and the baseline model.

3. The predicted depth maps were processed to convert disparities into depths, making them comparable to the ground truth.

4. Mean Squared Error (MSE) was calculated between the predicted depths and the ground truth for both models.

The evaluation metrics were compiled and analyzed to quantitatively assess the performance of each model. Additionally we qualitatively assess the performance of our model by observing the depth predictions against the ground truth as well as the error differences.

3.4 Instance Seg. Eval. Setup

We used the mean average precision metrics from the COCO Challenge (Lin et al., 2014), including AP, AP50, and AP75, to compare the performance of our fine-tuned Mask DINO against Prompter's original module.

4 Proposed Model

We use Depth Anything (Yang et al., 2024) as our depth estimation model and Mask DINO (Li et al., 2023a) as our instance segmentation engine.

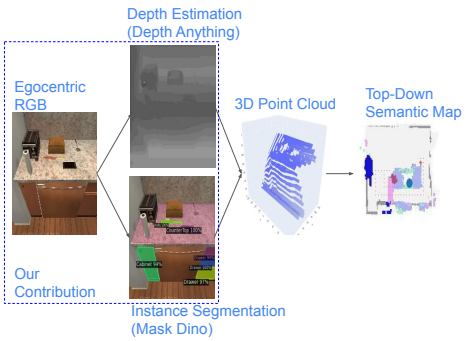


Figure 1: Visual Processing Pipeline. We use Depth Anything for depth estimation and Mask Dino for Instance Segmentation.

4.1 Depth Estimation

We incorporated the "Depth Anything" model, a pre-trained foundation model designed for robust monocular depth estimation. A significant challenge we encountered during the fine-tuning process was the model's inherent computation of disparity rather than depth. This fundamental difference led to difficulties in adjusting the scale to compute relative depth accurately. Various methods we attempted to apply in order to transform disparity into a usable depth format struggled under a typical mean squared error (MSE) loss, largely due to scale discrepancies. Qualitatively looking at the images we observed that the optimization process was further complicated by the model producing predominantly black images, indicating a failure in adapting the transformations required for our specific dataset. This issue suggested a misalignment between the model's output and the expected range of depth values within our application, leading to substantial difficulties in achieving meaningful optimization.

Despite these challenges, the strong generalization capability of the pre-trained "Depth Anything" model across a wide range of conditions and its robust performance as documented in the work by Yang et al. ultimately made it suitable for our purposes in its original, unmodified form. Utilizing the model without further fine-tuning allowed us to leverage its advanced capabilities and avoid the extensive computational resources typically required for training from scratch. The foundation model's proficiency in handling diverse imaging conditions proved to be important, providing reliable depth estimations across various scenes in our dataset without the need for extensive retraining.

4.2 Instance Segmentation

We used Mask DINO with a pre-trained ResNet50 backbone for instance segmentation. We trained a Mask DINO semantic segmentation head with the relevant 95 classes that FILM's segmentation engine requires and fine-tune the backbone with one-tenth learning rate. We used a confidence level threshold of 0.5 to filter out inaccurate segmentation masks. The segmentation mask is trained on a subset of our collected dataset. Specifically, the subset is created by a one-out-of-ten equal-spacing subsampling of 1,637 episodes, which amounts to 48,571 images with ground truth annotations of detected objects, including 770,255 total detected objects. We trained Mask DINO for 70,000 iterations of images (i.e. 1.44 epochs of our subset) for evaluations.

5 Baselines

5.1 Unimodal Baselines

In the Embodied Instruction Following (EIF) task setting, a purely unimodal baseline is insufficient. Language is needed for goal-setting and a visual stream is needed for navigation. However, the Seq2Seq and Episodic Transformer (ET) models, evaluated in scenarios focusing solely on either visual or linguistic input, shed light on the unique challenges and limitations inherent to each modality within EIF tasks.

5.1.1 Seq2Seq Unimodal Variants

Seq2Seq, initially conceptualized for language translation, adapts to EIF by processing input to predict actions. Its unimodal variants include:

- **Visual-Only Seq2Seq:** This variant leans entirely on visual data to navigate and perform tasks, omitting linguistic guidance. The model's performance in this setting highlights the pivotal role of linguistic instructions in providing context and objectives for tasks, often resulting in suboptimal performance when operating with visual data alone.
- **Language-Only Seq2Seq:** Operating without visual inputs, this version attempts to rely entirely on textual instructions. This approach

tests the model's capability to comprehend and strategize task execution based purely on linguistic input, underscoring the challenge of precise action prediction and environmental interaction in the absence of visual context.

5.1.2 Episodic Transformer Unimodal Variants

ET employs a transformer architecture for a comprehensive processing of inputs. Its unimodal adaptations are designed to evaluate the efficacy of singular modalities:

- **Visual-Only ET:** Focusing solely on visual inputs, this version aims to understand and navigate the environment. The limitations of this approach become apparent in tasks that require detailed instructions or interactions not discernible through visual analysis alone, indicating the necessity for linguistic data integration.
- **Language-Only ET:** By excluding visual cues, this model evaluates the sufficiency of textual instructions for task completion. This setup reveals the challenges in navigating and interacting with the environment without the aid of visual information, emphasizing the importance of multimodal inputs for effective task execution.

As shown in our previous report, these Unimodal baselines unilaterally achieve a 0% success rate. This finding is excellent in that it shows us that for meaningful results, both visual and language modalities are needed. However, it also shows that meaningful analysis beyond a modality's necessity requires a multimodal approach.

5.2 Simple Multimodal Baselines

The Seq2Seq and ET models also serve as baselines for basic multimodal integration, highlighting the initial steps towards combining visual and linguistic data to inform action prediction.

5.2.1 Seq2Seq

For EIF tasks, Seq2Seq employs a Convolutional Neural Network (CNN) to encode visual information and a Long Short-Term Memory (LSTM) network for processing linguistic instructions. This method directly integrates these inputs to predict a sequence of actions, demonstrating an effective end-to-end solution for multimodal integration.

5.2.2 Episodic Transformer (ET)

ET enhances the Seq2Seq approach by incorporating a multimodal transformer architecture that consolidates visual observations with linguistic instructions over the course of interactions. Unlike Seq2Seq, ET is designed to maintain a comprehensive history of inputs and actions, facilitating a more nuanced understanding and prediction of future actions. This longer context length, per se, allows for longer-term strategies for navigation and environment interaction.

5.3 Complex Multimodal Baselines

In exploring the FILM and Prompter frameworks, our ablation studies focus on key components that influence spatial reasoning, linguistic processing, and environmental interaction. Below are the specific parameters varied in our studies, outlined to facilitate understanding of their roles and configurations.

5.3.1 Centering Strategy

- **Local Adjustment** (for Prompter): Adapts interaction positions based on local environmental feedback, aiming to enhance Prompter's interaction efficiency.
- **Simple** (for FILM): Utilizes a straightforward approach to interaction positioning without dynamic adaptation, serving as a comparative baseline for Prompter's local adjustment strategy.

5.3.2 Language Processing

- **Granular Text (GT)**: Both models employ detailed linguistic inputs to ensure that the focus remains on the interplay between linguistic instructions and other model components.

5.3.3 Depth Perception

- **Learned Depth**: Incorporates algorithms to autonomously interpret spatial relationships, enhancing the model's navigational accuracy.
- **Ground Truth Depth**: Relies on exact depth information from the environment, bypassing the need for depth learning to simplify spatial understanding tasks.

5.3.4 Semantic Policy Generation

- **Masked Language Modeling (MLM) for Prompter**: Enhances language understanding and semantic decision-making through the use of MLM techniques.
- **Convolutional Neural Network (CNN) for**

Table 1: Feature Comparison of EIF Methods

Features / Methods	Seq2Seq	E.T.	FILM	Prompter
End-to-End	X	X		
Modular Design			X	X
Uses Semantic Maps			X	X
LLM-enabled				X

FILM: Applies CNNs for generating semantic segmentation and policy decisions, providing a foundation for assessing MLM's impact in Prompter.

5.3.5 Semantic Segmentation

- **Ground Truth**: Utilizes precise, manually verified segmentation data to provide a high accuracy benchmark.
- **Learned Segmentation**: Assesses the model's performance using segmentation generated through learning algorithms, focusing on the effectiveness of autonomous environmental understanding.

These parameters delineate the scope of our ablation studies, aiming to isolate and evaluate the contributions of specific model features to the effectiveness of the FILM and Prompter frameworks in executing EIF tasks. More specifically, these ablation studies consisted of testing each combination of the aforementioned parameters. Given two potential settings for each the centering strategy, depth perception, semantic policy generation, and semantic segmentation, our study evaluated 16 different model configurations. Moreover, by methodically adjusting these model settings, we sought to elucidate the intricate dynamics between model architecture, linguistic integration, and spatial reasoning within complex task environments.

6 Results (1 page)

In analyzing the performance of our agent across different ablations, we performed four ablations. Prompter proports to improve the semantic policy and centrering stratagies used in FILM, we ablate over both these differences to find the optimal configuration of the model. We further ablate over ground truth segmentation / depth for a total of 16 ablations.

6.1 Quantitative Metrics

We measure Success Rate (SR), Action Success Rate (Action SR), and average steps for each ablation. SR indicates the overall success rate of an ablation, while Action SR measures the effectiveness of agent actions within the environment. Failures include scenarios like an agent blocked by an object or misusing tools (e.g., using a spoon to cut lettuce). An agent is terminated after 10 failures, emphasizing the importance of successful actions for episode success.

The average steps are further split by episode success, indicated which episodes fail due to timing out and which fail due to repeated action failure.

Furthermore, for our updated models, we perform a comparison between the models used in FILM/Prompter and our updated models by calculating the average MSE of both on the evaluation set we collected. This can be seen in Table 7.

For Instance Segmentation, we perform a comparison between the models used in FILM/Prompter and our updated models by calculating the Average Precision (AP) for both models on the evaluation set we collected. This can be seenin Table 8.

6.2 Qualitative Metrics

The qualitative metrics collected in this study encompass various elements related to model behavior and task outcomes across the 16 ablation studies. These include:

- **Task Descriptions and Failure Reasons**: A qualitative analysis of specific episodes and the reasons for failures, highlighting errors in object detection, interaction, and navigation.
- **Error Message Clusters**: Grouping of error messages based on content, providing insights into common error themes such as navigation obstacles, object identification, interaction issues, and system-related errors.
- **Qualitative Observations of Agent Behavior**: Observations of agent behavior during task execution, such as wandering off, misinterpreting objects, or struggling with spatial constraints.

- **Example Cases of Model Failures**: Detailed descriptions of cases where models failed to complete tasks, emphasizing common patterns and underlying causes of these failures.

These qualitative metrics were gathered to better understand the types of errors that occur in different ablation studies, the nature of agent behavior during task execution, and the broader patterns of performance challenges faced by the models. The collected metrics provide a comprehensive perspective on the qualitative factors that can impact overall success and guide further analysis and improvements in the model's performance.

7 Analysis (2 pages)

In this section, we present a comprehensive analysis of the 16 different ablation studies conducted on our Prompter/FILM. Each ablation focuses on a unique aspect of our model, aiming to understand its impact on the overall performance. The key metrics evaluated include success rate, action success rate, average steps per episode, steps in failure and success cases, and a detailed error analysis for each ablation study. Summarized in Tables 2, 3, and 4.

The ablation studies are divided into two categories: Local Adjustment (LA) and Simple (S), these ablate over the semantic policy used, S being FILMs semantic policy, and LA being Prompters policy.

7.1 Performance analysis

We begin by analyzing Table 2), which summarizes the outcomes of ablation studies, comparing CNN and MLM under Simple and Local Adjustment conditions. This analysis focuses on the use of Ground Truth and Learned models for instance segmentation and depth, examining how these affect Success Rates (SR) and Action Success Rates (Action SR). A key insight is the influence of the semantic policy, particularly with learned models, and the varying impact of Local Adjustment based on these semantic policies.

When using Ground Truth data, the Success Rates across both Simple and Local Adjustment setups remain consistently high, with minimal variation across different semantic policies. This suggests that the choice of policy has a negligible impact when Ground Truth data is used, indicating that accurate and reliable data can overcome differences in semantic approach. In particular, CNN with Local Adjustment shows a Success Rate of 96.09%, with Action Success Rates above 95%. Similarly, MLM achieves comparable success with Ground Truth data, showing 92.97% for Simple and 92.19% for Local Adjustment, again with high Action Success Rates. This highlights the robustness of these configurations when high-quality data is used.

However, when switching to learned models, the choice of semantic policy becomes a significant factor, leading to noticeable performance disparities. In the GT-S/L-D configuration, where the depth information is learned, Success Rates drop across the board, especially for CNN in the Simple setup (72.66%) and Local Adjustment (78.12%). Despite these declines, Action Success Rates remain high, demonstrating that while overall task success may wane, agents can still execute individual actions with a high degree of accuracy.

The Local Adjustment setup generally exhibits greater resilience when using learned models, especially with CNN, suggesting that this adjustment may help compensate for the reduced quality of learned data. However, the results indicate that the impact of Local Adjustment diminishes when using Ground Truth, emphasizing that semantic policy choice plays a more critical role when data reliability is compromised.

For the LL configuration, which represents fully learned setups, the Success Rates drop significantly, highlighting the challenges in achieving successful task completion with predicted data. Here, the choice of semantic policy is crucial, with CNN showing a Success Rate of 56.25% in Simple and 60.94% in Local Adjustment, while MLM has slightly higher rates (69.53% and 68.75%, respectively). Despite these challenges, Action Success Rates remain steady, pointing to the robustness of action execution, even in less reliable conditions.

Overall, the analysis underscores that the choice of semantic policy has the most significant impact when using learned models, with Local Adjustment showing a mitigating effect, particularly with CNN. When Ground Truth data is used, these differences diminish, highlighting the critical role of accurate data in ensuring consistent and high-performing task completion.

Practially, this means that for improved models, the differnces between Prompter and FILM should vanish and that the semantic policy proposed by Prompter largly only improves performance under imperfect depth / segmentation.

Table 2: Summary of Simple and Local Adjustment Ablation Studies for CNN and MLM - Ground Truth (GT), or Learned (L), Segmentation (S), and Depth (D)

	CNN				MLM			
	Simple		Local Adjustment		Simple		Local Adjustment	
Ablation	SR	Action SR	SR	Action SR	SR	Action SR	SR	Action SR
GT	89.84%	96.01%	96.09%	95.42%	92.97%	96.02%	92.19%	95.05%
GT-S/L-D	72.66%	94.68%	78.12%	94.52%	82.03%	95.00%	76.56%	95.00%
L-S/GT-D	58.59%	95.00%	67.97%	94.25%	63.28%	95.41%	68.75%	94.99%
LL	56.25%	94.78%	60.94%	93.77%	69.53%	95.43%	68.75%	95.22%

Table 3: Local Adjustment ablation studies - Ground Truth (GT) and Learned (L), Segmentation (S), and Depth (D) Average steps (AvS) of Success and Failure cases

	CNN			MLM		
Ablation	Avg. Steps	AvS Success	AvS Failure	Avg. Steps	AvS Success	AvS Failure
GT	141.27	121.80	620.2	148.40	115.53	536.2
GT-S/L-D	182.09	143.76	319.0	197.21	134.71	401.37
L-S/GT-D	330.68	184.08	641.76	323.81	173.08	655.43
LL	358.48	240.27	542.9	334.17	206.64	614.75

7.2 Analysis by number of steps

Local Adjustment Ablations (Table 3): The Ground Truth (GT) setup exhibited the most proficient performance, with lower average steps in both success and failure cases compared to learned configurations. Notably, the GT S/D configuration outperformed others, suggesting the critical role of accurate, real-world data in enhancing the model's effectiveness. On the contrary, the LL V/T configuration showed the highest average steps, indicating challenges in handling learned visual and textual inputs simultaneously.

Simple Ablations (Table 4): Simplifying the ablations presents an interesting contrast. Here, the GT configurations again demonstrate superior performance, reinforcing the value of ground truth data. However, the reduced complexity in Simple (S) ablations did not necessarily translate to lower average steps across all cases. We believe this shows the impact of environmental and task complexity on the agent's learning and decision-making processes.

These analyses underline the importance of accurate data and the complexity of interactions between the agent's perception and action mechanisms. While GT configurations consistently show promise, the variations in performance across different ablations show the need for further improvement into optimizing the learned to perform closer to it's ground truth components for enhanced agent performance.

In order to provide a more granular understanding of where the ablated models were failing, analysis was also conducted on the actions and objects for which the models most commonly failed its high-level tasks. An investigation into the problematic actions was first conducted, averaging across all ablations to account for model-specific factors.

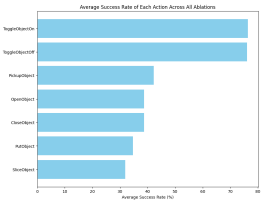


Figure 2: Task Success Rate Across Actions

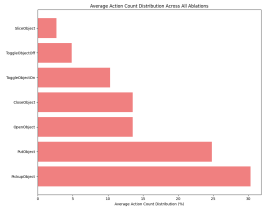


Figure 3: Frequency of Actions

Figure 2 shows the probability of completing each action. The results show that the model achieves excellent performance on simple actions, such as toggling objects. Conversely, the models performed extremely poorly on almost all of the other, more complicated tasks, such as slicing an object or putting it at some location. These findings indicate that the model performance could be improved by improving the model's mechanism of predicting and executing actions especially those that are more complicated. For example, the completion rate of high-level tasks could be improved by addressing the problems, such as agent-environment collisions, that are preventing the com-

Table 4: Simple ablation studies - Ground Truth (GT) and Learned (L), Segmentation (S), and Depth (D) Average steps (AvS) of Success and Failure cases

	CNN			MLM		
Ablation	Avg. Steps	AvS Success	AvS Failure	Avg. Steps	AvS Success	AvS Failure
GT	155.27	104.99	600.08	168.92	124.34	758.44
GT-S/L-D	218.16	165.24	358.80	196.20	136.51	468.65
L-S/GT-D	361.75	189.56	605.42	363.42	175.56	687.19
LL	375.66	192.19	611.54	329.34	192.12	642.49

pletion of complex actions.

In order to provide better context of these success rates, the frequency of the actions were also plotted and are given in figure 3. This figure further supports the need to address the issues with these more complicated tasks, such as "PutObject" or "PickupObject". This is because, despite being more error prone, these actions comprise a much larger portion of the agent's executed actions.

Similar analysis was also performed with action-objects, the objects with which an agent was tasked with completing its actions. The results shown in figure 4 show the probability of the model completing its high-level task(s) given the corresponding action-objects to be found. This figure interestingly shows that the model has a 0% success rate with a few objects, such as an "AppleSliced" or a "Drawer". From other analysis conducted on find rates of action-objects, it was observed that many the low-success action objects correspondingly have very low probabilities of being found by the model. These results indicate that the vision component of the model fails to detect certain classes of objects. Since the model can't complete a task with an object if it can't find it, this demonstrates that model performance could be improved by addressing issues in the vision component, such as visual synonymy (e.g,m "AppleSliced" vs. "Apple").

Figure 5, depicting the probability of action-objects found by the ablation models, supports this view. This is because all eight of the ablations in this figure have learned semantic-mapping components and considerably lower find rates than that of ground-truth semantic mapping methods (100%). Moreover, this demonstrates that by integrating a better vision component which finds action-objects with greater proficiency, our proposed model could improve upon the performance of current models. These results are shown in figure 5.

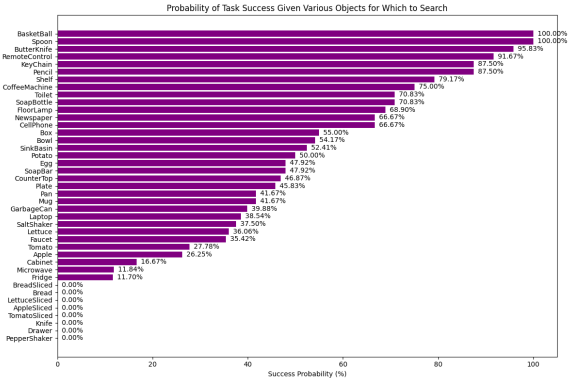


Figure 4: Probability of task success across action-objects

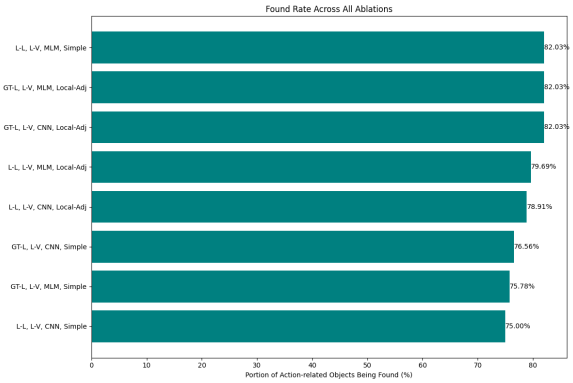


Figure 5: Percentage of action-objects found across ablations

7.3 Cluster Analysis of Error Messages

Looking into errors can help us identify if they are problematic. The only errors that are of interest are those which of which the probability of success is < 1. However, it is difficult to categorize our errors to determine which key words are similar to others manually. Therefore, we cluster our error messages using standard NLP tokenization and vectorize using TF-IDF.

Categorical label insights from NLP processing:

- **Cluster 0: Navigation and Obstruction Errors** This cluster appears to focus on

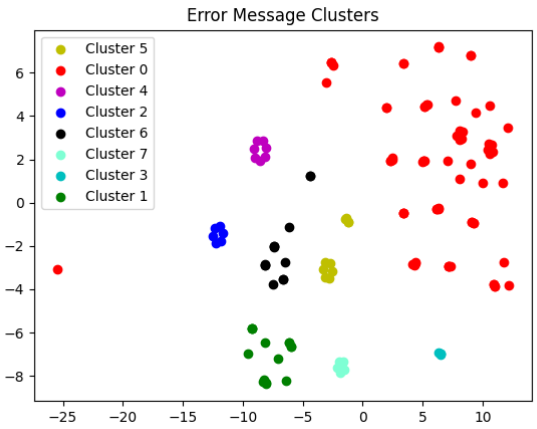


Figure 6: Visualization of error message clusters

movement-related errors, potentially involving the agent encountering obstacles. Keywords such as ”Moving”, ”blocking”, and specific object references like ”sidetable”, ”safe”, and ”chair” suggest scenarios where the agent's path is obstructed by furniture or objects, leading to navigation issues.

- **Cluster 1: Object Identification and Counting Errors** The presence of numbers alongside ”found”, ”object”, and ”apple” hints at errors related to object identification or counting tasks. This cluster might be capturing scenarios where the agent fails to accurately count or identify specific items, such as apples or cups, within the environment.

- **Cluster 2: Interaction and Recognition Errors** Keywords like ”Locate”, ”interact”, ”mask”, ”bad”, ”could”, and ”target” point to errors involving the agent's ability to recognize or interact with objects. It suggests difficulties in correctly identifying targets or executing interactions with them, possibly due to masking or bad recognition.

- **Cluster 3: Object-Object Action Execution Errors** With terms such as ”Knife”, ”holding”, ”slicing”, and ”agent”, this cluster likely represents errors related to specific action executions from one object to another, such as slicing object X with a knife. The keywords indicate challenges the agent faces in performing precise actions or holding items correctly.

- **Cluster 4: Physical Interaction and Collision Errors** This cluster signifies errors arising from physical interactions within the environment, like ”collision”, ”teleport”, and ”hand”. Errors might involve unintended collisions with objects or teleportation issues, affecting how the agent manipulates or moves around objects like basketballs or apples.

- **Cluster 5: Placement and Validity Errors** Keywords such as ”Valid”, ”found”, ”place”, ”positions”, and ”receptacle” suggest errors related to placing objects in valid positions or receptacles. It captures situations where the agent struggles to find a valid spot for object placement or encounters objects that are already placed.

- **Cluster 6: System and Reference Errors** This cluster focuses on technical errors, with terms like ”system”, ”instance”, ”exception”, ”nullreferenceexception”, indicating issues with system references, instances, or unhandled exceptions. These are likely non-environmental errors that occur at a system or code level.

- **Cluster 7: Validation and Identification Errors** With ”Invalid”, ”id”, ”object”, and references to numerical codes, this cluster could be highlighting errors related to object validation or ID errors. It might involve the agent incorrectly identifying objects or dealing with invalid object states.

7.4 Analysis of Success Probabilities

After clustering the error messages based on their textual content, we further analyzed each cluster to determine the average probability of success, assuming an error occurs. This analysis provides insight into which clusters contain errors that are more likely to be successfully mitigated or resolved. The average success probabilities for each cluster are as follows:

These success probabilities were calculated by aggregating the success rates of individual errors within each cluster, weighted by their frequencies. This approach allows us to understand not just the common themes within each cluster (as indicated by the top terms analysis) but also how likely errors within these themes are to be overcome.

Notably, Cluster 7 demonstrates the highest average success probability, but only contains 5 errors. This suggests that errors within this cluster, despite their occurrence, are generally more manageable

Cluster	Weight	Avg. Success Probability
0	71	0.41
1	35	0.07
2	26	0.34
3	14	0.00
4	17	0.26
5	20	0.47
6	23	0.33
7	15	0.59

Table 5: Average success probabilities for error message clusters

or less critical. Conversely, Cluster 1 shows an average success probability of 0.07, indicating that errors categorized here are particularly challenging to resolve successfully. This variance underscores the value of clustering in identifying areas of concern and potential focal points for improvement in error handling mechanisms.

7.5 Qualitative Analysis and Examples

We have selected 5 representative episodes for qualitative analysis, including failure cases for all the 16 ablations, and displayed these samples in Table 6. In each of the episodes, the task is separated into subgoals of required actions, defined as a list of tuples, each tuple containing the object to interact with and the type of interaction. We analyzed the behavior of each ablations in more details by examining the detailed trajectory logs and the respective RGB inputs and produced segmentation maps, and we will discuss the more fruitful findings by each episode.

7.5.1 Episode 0

In Episode 0, the task is to find the CellPhone and examine it under the light of FloorLamp. The required actions are defined as [('CellPhone', 'PickupObject'), ('FloorLamp', 'ToggleObjectOn')].

In *L-S/GT-D FILM S* and *LL V/T FILM S*, FloorLamp was successfully detected, but the object is not yet reachable. The semantic policy failed to plan accordingly to approach the object and interact with it, instead wondering off and attempting to find the object again, but then identified a desk lamp as FloorLamp, shown in Figure 7, which they failed to interact with and got stuck on. This problem of semantic search policy only occurs in FILM without local adjustment, so this problem might be fixed by the local adjustment centering strategy to properly interact with the FloorLamp during the first successful detection.

Figure 7: The desk lamp shown in the upper left corner is wrongly detected as the FloorLamp in the task. The model's confusion between the FloorLamp and a desk lamp is one of the many visual synonymy problems the existing methods encounter, and demonstrates the bottleneck of the visual segmentation in the ALFRED task.

In *L-S/GT-D Pmpt S*, the vision pipeline wrongly recognized a part of the TV drawer as a CellPhone, and kept trying to interact with it, as shown in Figure 8, resulting in the failure. This particular example insisted on picking up this non-existant CellPhone, and while some other setups do encounter the same incorrect visual detection, they moved on after several failed interactions and finally succeeded.

7.5.2 Episode 7

In Episode 7, the task is to wash a bowl and put it into a cabinet, and the required actions for the task are defined as [('Bowl', 'PickupObject'), ('SinkBasin', 'PutObject'), ('Faucet', 'ToggleObjectOn'), ('Faucet', 'ToggleObjectOff'), ('Bowl', 'PickupObject'), ('Cabinet', 'OpenObject'), ('Cabinet', 'PutObject'), ('Cabinet', 'CloseObject')].

In *GT S/D FILM S*, *GT S/D Pmpt S*, *GT-S/L-D FILM S*, and *GT-S/L-D FILM LA*, the agent successfully finds the cabinet and opens it, but failed to put the bowl into the cabinet due to spacing issues. We can observe that this problem mostly happens in runs without local adjustment, highlighting the importance of the centering strategy, while also suggesting that there is still room for improvements.

Ablation	Episode 0	Episode 7	Episode 19	Episode 48	Episode 110
GT S/D FILM S	Success.	Failure, no place to put object.	Success	Failure, object failed open/close.	Failure, hand object collision.
GT S/D Pmpt S	Success.	Failure, no place to put object.	Success.	Success.	Failure, timeout at 1000 steps.
L-S/GT-D FILM S	Failure, timeout at 1000 steps.	Failure, cabinet not opened.	Failure, can't slice object without knife.	Failure, timeout at 1000 steps.	Failure, hand object collision.
L-S/GT-D Pmpt S	Failure, object ID error.	Failure, cabinet not opened.	Failure, can't slice object without knife.	Failure, timeout at 1000 steps.	Failure, object not found.
GT-S/L-D FILM S	Success.	Failure, no place to put object.	Failure, timeout at 1000 steps.	Success.	Failure, Object ID invalid.
GT-S/L-D Pmpt S	Success.	Success.	Failure, can't slice object without knife.	Success.	Failure, object not found.
LL S/D FILM S	Failure, timeout at 1000 steps.	Failure, blocked by chair.	Failure, GameObject blocks action.	Failure, timeout at 1000 steps.	Failure, cup can't be sliced.
LL S/D Pmpt S	Success.	Success.	Failure, GameObject blocks action.	Failure, timeout at 1000 steps.	Failure, object not found.
GT S/D FILM LA	Success.	Success.	Success	Success.	Failure, timeout at 1000 steps.
GT S/D Pmpt LA	Success.	Failure, blocked by cabinet.	Success.	Success.	Failure, timeout at 1000 steps.
L-S/GT-D FILM LA	Success.	Failure, no error message.	Failure, timeout at 1000 steps.	Failure, timeout at 1000 steps.	Failure, hand object collision.
L-S/GT-D Pmpt LA	Success	Success.	Failure, blocked by counter.	Failure, timeout at 1000 steps.	Failure, hand object collision.
GT-S/L-D FILM LA	Success.	Failure, no place to put object.	Success.	Failure, bad interact mask.	Failure, timeout at 1000 steps.
GT-S/L-D Pmpt LA	Success.	Success.	Failure, timeout at 1000 steps.	Success.	Failure, timeout at 1000 steps.
LL S/D FILM LA	Failure, blocked by FP219:Cube.	Success.	Failure, GameObject blocks action.	Failure, timeout at 1000 steps.	Failure, blocked by chair.
LL S/D Pmpt LA	Success.	Success.	Failure, GameObject blocks action.	Failure, timeout at 1000 steps.	Failure, object not found.

Table 6: Samples of failure cases for FILM (FILM) and Prompter (Pmpt) Ground Truth (GT) and Learned (L), Segmentation (S), and Depth (D) with Local Adjustment (LA) or Simple (S) ablations.

In *L-S/GT-D FILM S* and *L-S/GT-D Pmpt S*, the visual pipeline wrongly detected the dishwasher as the cabinet, resulting in the "cabinet not opened" error, again highlighting the bottleneck nature of visual segmentation in the ALFRED task.

In *LL FILM S* and some successful runs, we found that the agent attempted to put the bowl into the kitchen sink from the other side of the kitchen counter (backside of the faucet) as the sink would be within the valid interaction range, but there are two chair objects in the way. This observation is repeatedly found in tasks in the kitchen involving interactions with the sink, and we believe it is the primary cause of the chair failures. This should be a limitation of the ALFRED task, as this action is perfectly reasonable in real-life scenarios.

7.5.3 Episode 19

In Episode 19, the task is a more complicated, involving cutting a tomato with a knife, put the knife in the sink, then put the sliced tomato into the fridge. The required actions are defined as [('Knife', 'PickupObject'), ('Tomato', 'SliceObject'), ('SinkBasin', 'PutObject'), ('TomatoSliced', 'PickupObject'), ('Fridge', 'OpenObject'), ('Fridge', 'PutObject'), ('Fridge', 'CloseObject'), ('Fridge', 'OpenObject'), ('TomatoSliced', 'PickupObject'), ('Fridge', 'CloseObject'), ('CounterTop', 'PutObject')].

In most of the failed runs, we found that when the agent is performing the first action of picking up the knife, it picks up a cooking shovel instead due to the visual segmentation incorrectly detected it as the knife. This resulted in a cascade of various different failures later on with different runs, while most runs with *GT S* finishes with success as they picked up the correct knife. This observation once again highlighted that the visual segmentation accuracy is the bottleneck for the ALFRED task. Additionally, some of the runs will ignore the "can't slice object without knife" error, and proceeds with the next actions without cutting the tomato, resulting in failure of finding TomatoSliced later on. Collectively, this exposed the limitation of the FILM/Prompter pipeline, as they do not have mechanisms to backtrack to the last action once an action is considered done.

Figure 8: A part of the TV drawer is wrongly detected as the CellPhone object in the task.

Figure 9: The sliced lettuce is detected as multiple lettuce instead of one sliced lettuce.

7.5.4 Episode 48

In Episode 48, the task is to cut a lettuce similarly to Episode 19, but then take the sliced lettuce out of the fridge and put it into the garbage can. The required actions are defined as [('Knife', 'PickupObject'), ('Lettuce', 'SliceObject'), ('SinkBasin', 'PutObject'), ('LettuceSliced', 'PickupObject'), ('Fridge', 'OpenObject'), ('Fridge', 'PutObject'), ('Fridge', 'CloseObject'), ('Fridge', 'OpenObject'), ('LettuceSliced', 'PickupObject'), ('Fridge', 'CloseObject'), ('GarbageCan', 'PutObject')].

In these runs, we observed the same error of picking up a cooking shovel instead of a knife as that of Episode 19. Additionally, as shown in Figure 9, the correctly sliced lettuce is detected as multiple different lettuces by the visual segmentation, resulting in never finding the object LettuceSliced. In addition to highlighting the segmentation being the bottleneck of the ALFRED task, this suggests that using only a pre-trained segmentation model may not be sufficient for the ALFRED task, as specific objects like LettuceSliced in contrast to Lettuce may be indifferentialable by general segmentation models.

7.5.5 Episode 110

In Episode 110, the task is similar to that of Episode 48, minus putting the sliced lettuce into the fridge and added washing the sliced lettuce. The required actions are defined as [('Knife', 'PickupObject'), ('Lettuce', 'SliceObject'), ('SinkBasin', 'PutObject'), ('LettuceSliced', 'PickupObject'), ('SinkBasin', 'PutObject'), ('Faucet', 'ToggleObjectOn'), ('Faucet', 'ToggleObjectOff'), ('LettuceSliced', 'PickupObject'), ('GarbageCan', 'PutObject')].

In this example, every run failed. We observe that in most runs, the agent successfully picks up the knife, but failed to find the lettuce from the image. Some of them mistakenly recognize a green cup in the sink as the lettuce and attempts to cut it, while others roam around without ever successfully finding it. This observation is especially interesting as the runs with *GT-S* also fails to find the lettuce. We further looked through every RGB frames in the agent's trajectory, and were not able to find a lettuce within the agent's segmentation either. We further examined the run with the highest overall success rate *GT S/D FILM LA*, and found that in all its 5 failure runs within the total 128 runs, 4 of them describe a task as Episode 110, and all failed with finding lettuce, suggesting that this error run may be caused by a mistake either in the ALFRED dataset or from its backend engine procThor.

7.6 Analysis of New Models

7.6.1 Depth Anything

Our results show an improvement in depth prediction accuracy when using the "Depth Anything" model over the baseline prompter depth model. The mean MSE for the "Depth Anything" model was 0.052, while the baseline model achieved a mean

MSE of 0.087, indicating an increased performance when using the "Depth Anything" model.

Model	Mean MSE	Improvement
Prompter's Depth	0.087	-
Depth Anything	0.052	40.23%

Table 7: Comparison of MSE values between the baseline depth model and the "Depth Anything" model.

The improvement in MSE, quantified as approximately 40.23%, indicates that the "Depth Anything" model achieves a better MSE in depth estimation compared to the baseline model when computing this across the ground truth depth predictions. This enhancement can be attributed to the advanced learning mechanisms and robust data handling capabilities of the "Depth Anything" model.

7.6.2 Mask DINO

The results on the validation set are shown in Table 8. Due to the limitation of the size of our fine-tuning dataset and that we used the same model for both small and large objects, our fine-tuned Mask DINO is outperformed by Prompter's original instance segmentation module.

Methods	AP	AP50	AP75
Prompter	88.232	90.759	68.709
Mask DINO	41.275	67.035	41.118

Table 8: AP, AP50, and AP75 of Prompter's original instance segmentation versus the fine-tuned Mask DINO on validation set.

We further investigated the performance of our fine-tuned Mask DINO by evaluating on a subset of the training set which our Mask DINO was not trained on. The reason for this additional evaluation is that we were not able to obtain enough samples from the validation set to obtain at least one sample for each of the target classes, thus resulting in NaN per-class average precision for 31 out of 95 classes, which is insufficient for our analysis of the performance of our model. We show the evaluation results in Figure 10. We can observe an overall better average performance compared to the validation set, showing that our model performs better for the classes not included in the current validation set. Additionally, we can also observe that our model mainly performs poorly on the small objects, while Prompter used two separate models for small and large objects.

AP	AP50	AP75	APs	APm	APl
55.116	82.135	58.711	43.673	84.711	91.086

[04/27 17:36:07] d2.evaluation.coco_evaluation INFO: Per-category segm AP:

category	AP	category	AP	category	AP
AlarmClock	72.572	Apple	55.360	ArmChair	73.127
BaseballBat	25.653	BasketBall	75.137	Bathtub	61.347
Bed	75.502	Book	66.246	Bowl	61.598
Box	72.313	Bread	59.628	ButterKnife	57.316
CD	53.823	Cabinet	78.345	Candle	48.388
Cart	23.295	CellPhone	64.775	Cloth	56.873
CoffeeMachine	83.234	CoffeeTable	52.976	CounterTop	53.781
CreditCard	47.341	Cup	58.174	Desk	41.329
DeskLamp	69.503	DiningTable	52.272	DishSponge	55.998
Drawer	65.669	Dresser	49.103	Egg	48.353
Faucet	31.878	FloorLamp	53.070	Fork	16.652
Fridge	79.734	GarbageCan	78.615	Glassbottle	52.240
HandTowel	77.383	HandTowelHolder	29.257	HousePlant	31.087
Kettle	53.214	KeyChain	24.695	Knife	44.941
Ladle	26.623	Laptop	73.942	LaundryHamper	65.546
LaundryHamperLid	54.266	Lettuce	63.684	LightSwitch	67.454
Microwave	88.145	Mug	61.721	Newspaper	58.992
Ottoman	75.891	Pan	60.603	PaperTowelRoll	58.651
Pen	57.933	Pencil	23.908	PepperShaker	41.174
Pillow	74.073	Plate	60.389	Plunger	45.535
Pot	70.502	Potato	46.746	RemoteControl	62.214
Safe	73.690	SaltShaker	40.370	ScrubBrush	39.767
Shelf	53.706	ShowerDoor	59.218	SideTable	39.358
Sink	54.398	SoapBar	60.009	SoapBottle	60.331
Sofa	71.829	Spatula	17.531	Spoon	30.570
SprayBottle	57.403	Statue	37.155	StoveBurner	30.372
StoveKnob	47.499	TVStand	54.073	TeddyBear	70.464
Television	81.688	TennisRacket	25.246	TissueBox	62.869
Toaster	78.208	Toilet	85.493	ToiletPaper	62.304
ToiletPaperHanger	19.727	Tomato	58.904	Towel	83.573
TowelHolder	16.720	Vase	59.073	Watch	19.879
WateringCan	50.996	WineBottle	63.820		

Figure 10: Mean Average Precision of Mask DINO evaluated on unseen training set. Notice APm and APl (for middle and large objects) achieve similar performance as Prompter's instance segmentation, but APs (for small objects) performs relatively poorly.

7.7 Qualitative Analysis

7.7.1 Depth Anything

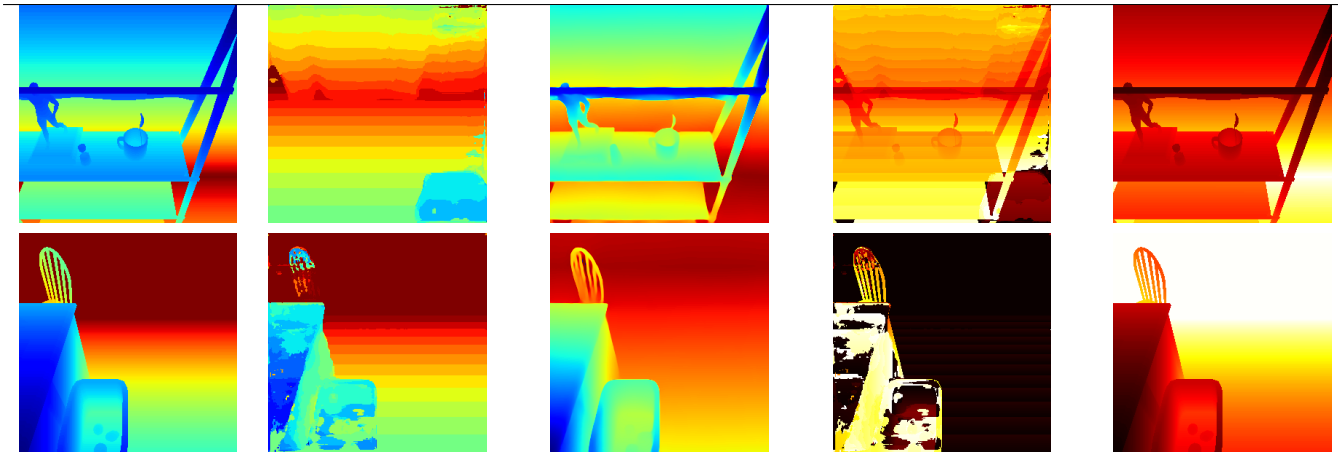
To visually represent the model performance, we selected three episodes (300, 1250, 1550) to showcase the depth prediction quality. Table **??** illustrates the performances qualitatively.

Figure 9 contains visualizations help to qualitatively validate the numerical findings, offering a clear visual understanding of how the "Depth Anything" model appears to have a clearer depth prediction when compared to the prompter's current depth model.

7.7.2 Mask Dino

We visualize the instance segmentation results for comparison in Table 10. As seen in the ground truth, there are often scenes where objects are placed very close to the camera frame, objects for the same class appear right next to each other, and only partial views of objects are collected. Additionally but not perhaps featured in the visualizations, there are objects are with different sizes, seen in different lightning conditions, and occluded by other objects. The Prompter baseline segmentation model performs well in some situation where our undertrained model does not. These situations

Qualitative Comparison Between Ground Truth, Depth Anything, and Baseline Prompter Depth.



Ground Truth	Prompter Depth	Depth Anything Depth	Prompter Error	Depth Anything Error

seem to be when instances of the same class are next to each other or when objects seem blend into the background. However, our model seems to do better at recognizing objects with partial views, which is essential for locating unseen objects during navigation. If the segmentation model can better detect objects in partial views, this will likely have a direct positive effect on success rate as the search policy will be able to switch to acting on the object instead of missing it.

8 Conclusion

The development of embodied agents, particularly in the context of Embodied Instruction Following (EIF), is crucial for advancing AI's ability to interact with real-world and simulated environments. Our investigation focused on the Prompter and FILM methods, assessing their performance on EIF tasks within the ALFRED dataset. The evaluation included an updated vision pipeline, with a detailed analysis of segmentation and depth-based tasks, revealing several key insights.

Firstly, our experiments showed that the use of ground-truth data for depth and segmentation significantly boosts task success rates, with minor variations across different semantic policies. However, when transitioning to learned models, semantic policies play a more critical role in determining performance. The Local Adjustment setup generally exhibits greater resilience when using learned models, especially with CNN, indicating its potential in compensating for reduced data quality.

Secondly, by integrating advanced models like DepthAnything and MaskDino, we observed improved accuracy in depth estimation and competitive, albeit lower, performance in segmentation. The "Depth Anything" model achieved a 40% improvement in mean square error (MSE), while the fine-tuned MaskDino model demonstrated challenges in handling smaller objects compared to Prompter's existing instance segmentation. Despite these improvements, issues with action execution and object detection remain, emphasizing the need for further refinement in visual pipelines and semantic policies.

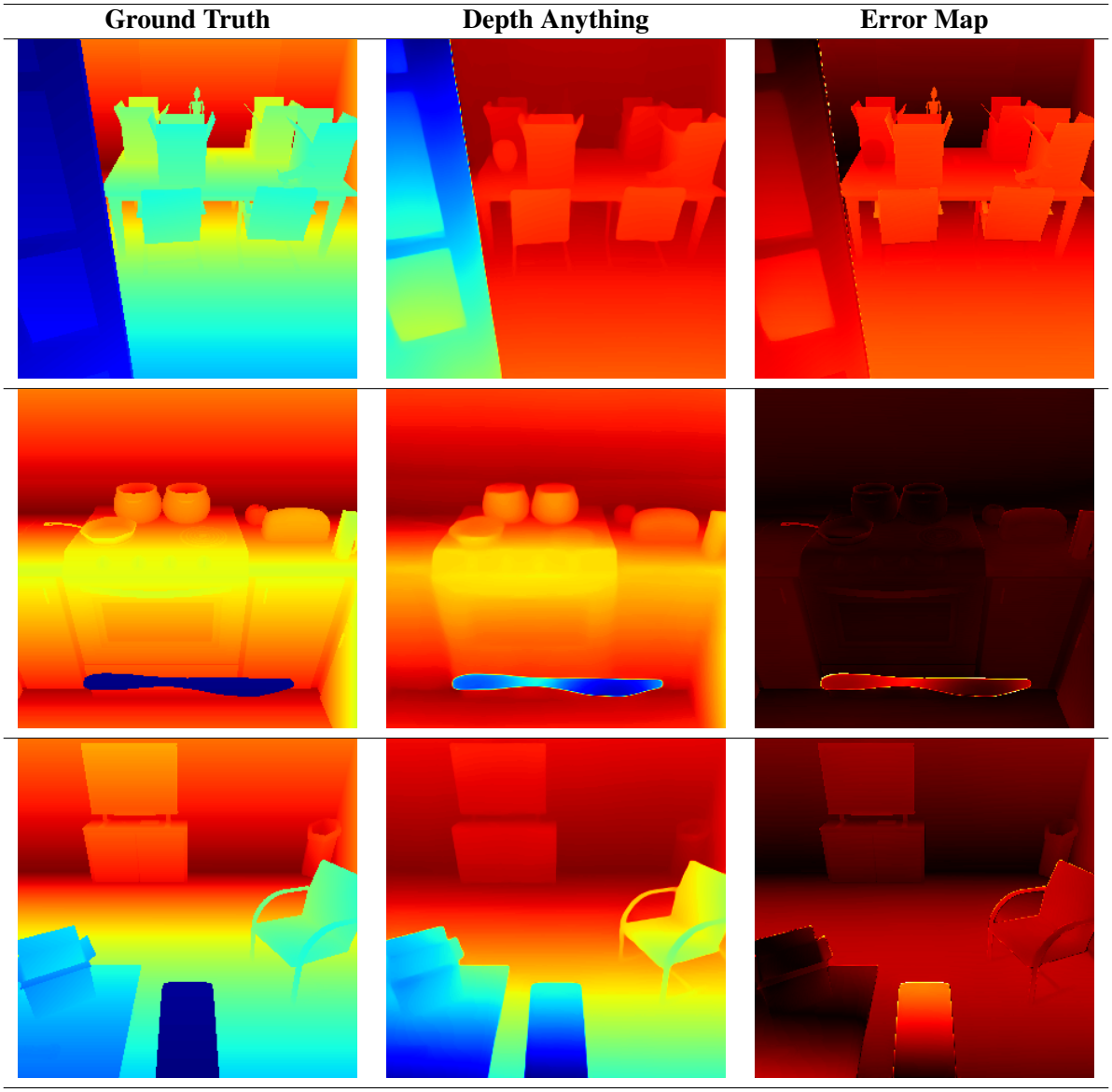
Overall, our analysis highlights the importance of accurate data and robust vision components in ensuring consistent performance in complex EIF tasks. The discrepancies between ground-truth and learned models underscore the need for continued research into improving learned components to achieve results on par with ground-truth setups.

9 Future work and Limitations (0.5-1 page)

Our segmentation model was fine-tuned on a small subset of the full training data and trained for a limited period, leading to a potential limitation in its generalization capacity. Additionally, the baseline uses two models for segmentation—one for small objects and one for large objects. Future work could explore using multiple models to improve the performance of our segmentation approach, especially for objects of varying sizes and under different lighting conditions.

Regarding depth estimation, our depth models were not fine-tuned in the ALFRED environment, relying solely on pre-trained monocular depth pre-

Table 9: Comparison of Ground Truth, Depth Anything Predicted Depth, and Error Maps



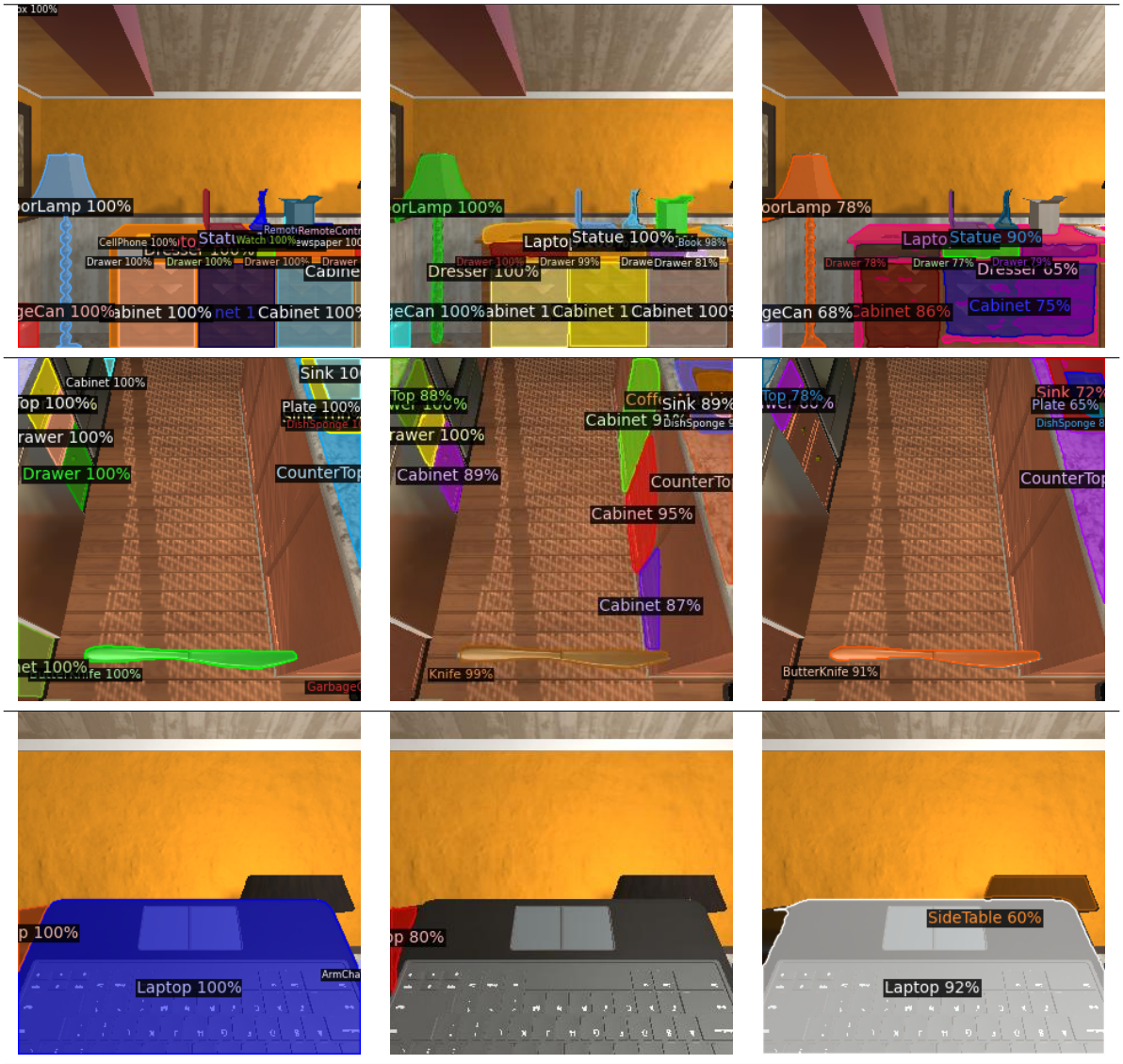
diction. Fine-tuning on ground-truth metric depth in this environment could improve performance. Monocular depth prediction has inherent limitations, as it relies on single RGB frames to estimate depth without multi-view references. A more theoretically robust approach would involve multi-view depth estimation, offering a more accurate and reliable depth measurement. Additionally, our use of the DepthAnything model, which produces disparity, poses a challenge when converting disparity into depth, leading to potential accuracy issues.

Beyond segmentation and depth, future work could focus on enhancing semantic policies. The current semantic policy lacks flexibility when errors occur or when tasks require backtracking. Improving semantic policy resilience could lead to more robust agent behavior, especially in complex tasks involving dynamic interactions.

Another area for further research involves foundational models and unsupervised learning techniques. Our baseline models benefited from using pre-trained foundational models. Future research could investigate the impact of incorporating larger and more diverse data sources for unsupervised learning, aiming to improve adaptability and generalization in real-world scenarios.

Furthermore, addressing limitations in task execution and object interaction could provide significant performance gains. Our qualitative analysis revealed that some models struggle with specific actions or tasks, suggesting a need for improved action planning and error handling mech-

Table 10: Comparison of Instance Segmentation of Ground Truth, Prompter, and Mask DINO

Ground Truth	Prompter	Mask DINO
		

anisms. Efforts to refine these aspects could lead to more reliable and consistent performance in embodied instruction-following tasks, especially in real-world applications.

10 Ethical Concerns and Considerations

While embodied robotics, involving robots with physical forms interacting in real-world environments, typically raises significant ethical considerations, this paper does not delve into specific ethical issues. Instead, it focuses on technological advancements aimed at improving the capabilities and efficiency of embodied robotics. The primary concern here is enhancing the design and functionality of these robots without addressing the broader ethical implications such as job displacement, privacy concerns, or the moral aspects of autonomous decision-making. The paper assumes that existing ethical frameworks are sufficient to guide the development and application of these technologies.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Sarthak Bhagat, Simon Stepputtis, Joseph Campbell, and Katia Sycara. 2023. Sample-efficient learning of novel visual concepts. *arXiv preprint arXiv:2306.09482*.

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

Ting-Rui Chiang, Yi-Ting Yeh, Ta-Chung Chi, and Yau-Shian Wang. 2021. Are you doing what i say? on modalities alignment in alfred. *arXiv preprint arXiv:2110.05665*.

Mingyu Ding, Yan Xu, Zhenfang Chen, David Daniel Cox, Ping Luo, Joshua B Tenenbaum, and Chuang Gan. 2023. Embodied concept learner: Self-supervised learning of concepts and mapping through instruction following. In *Conference on Robot Learning*, pages 1743–1754. PMLR.

Haoran Geng, Songlin Wei, Congyue Deng, Bokui Shen, He Wang, and Leonidas Guibas. 2023. Sage: Bridging semantic and actionable parts for generalizable articulated-object manipulation under language instructions. *arXiv preprint arXiv:2312.01307*.

Qiao Gu, Alihusein Kuwajerwala, Sacha Morin, Krishna Murthy Jatavallabhula, Bipasha Sen, Aditya Agarwal, Corban Rivera, William Paul, Kirsty Ellis, Rama Chellappa, et al. 2023. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. *arXiv preprint arXiv:2309.16650*.

Rishi Hazra, Brian Chen, Akshara Rai, Nitin Kamra, and Ruta Desai. 2023. Egotv: Egocentric task verification from natural language task descriptions. *arXiv preprint arXiv:2303.16975*.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. 2023. Cogagent: A visual language model for gui agents. *arXiv preprint arXiv:2312.08914*.

Zhening Huang, Xiaoyang Wu, Xi Chen, Hengshuang Zhao, Lei Zhu, and Joan Lasenby. 2023. Openins3d: Snap and lookup for 3d open-vocabulary instance segmentation. *arXiv preprint arXiv:2309.00616*.

Yuki Inoue and Hiroki Ohashi. 2022. Prompter: Utilizing large language model prompting for a data efficient embodied instruction following. *arXiv preprint arXiv:2211.03267*.

Ayush Jain, Pushkal Katara, Nikolaos Gkanatsios, Adam W Harley, Gabriel Sarch, Kriti Aggarwal, Vishrav Chaudhary, and Katerina Fragkiadaki. 2024. Odin: A single model for 2d and 3d perception. *arXiv preprint arXiv:2401.02416*.

Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. 2023. Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. *arXiv preprint arXiv:2312.02126*.

Byeonghwi Kim, Jinyeon Kim, Yuyeong Kim, Cheolhong Min, and Jonghyun Choi. 2023. Context-aware planning and environment-aware memory for instruction following embodied agents. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10936–10946.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. *arXiv preprint arXiv:2304.02643*.

Aditya Kusupati, Gantavya Bhatt, Aniket Rege, Matthew Wallingford, Aditya Sinha, Vivek Ramanujan, William Howard-Snyder, Kaifeng Chen, Sham Kakade, Prateek Jain, et al. 2022. Matryoshka representation learning. *Advances in Neural Information Processing Systems*, 35:30233–30249.

Feng Li, Hao Zhang, Huaizhe Xu, Shilong Liu, Lei Zhang, Lionel M Ni, and Heung-Yeung Shum. 2023a. Mask dino: Towards a unified transformer-based framework for object detection and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3041–3050.

Xiaoqi Li, Mingxu Zhang, Yiran Geng, Haoran Geng, Yuxing Long, Yan Shen, Renrui Zhang, Jiaming Liu, and Hao Dong. 2023b. Manipllm: Embodied multimodal large language model for object-centric robotic manipulation. *arXiv preprint arXiv:2312.16217*.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer.

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

Tuomas Oikarinen and Tsui-Wei Weng. 2022. Clip-dissect: Automatic description of neuron representations in deep vision networks. *arXiv preprint arXiv:2204.10965*.

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.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR.

Gabriel Sarch, Yue Wu, Michael J Tarr, and Katerina Fragkiadaki. 2023. Open-ended instructable embodied agents with memory-augmented large language models. *arXiv preprint arXiv:2310.15127*.

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.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. ALFWorld: Aligning Text and Embodied Environments for Interactive Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Chan Hee Song, Jiaman Wu, Clayton Washington, Brian M Sadler, Wei-Lun Chao, and Yu Su. 2023. Llm-planner: Few-shot grounded planning for embodied agents with large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2998–3009.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, et al. 2023. Cogvlm: Visual expert for pretrained language models. *arXiv preprint arXiv:2311.03079*.

Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. 2024. Depth anything: Unleashing the power of large-scale unlabeled data. In *CVPR*.

Yunhan Yang, Xiaoyang Wu, Tong He, Hengshuang Zhao, and Xihui Liu. 2023. Sam3d: Segment anything in 3d scenes. *arXiv preprint arXiv:2306.03908*.

Renos Zabounidis, Joseph Campbell, Simon Stepputtis, Dana Hughes, and Katia P Sycara. 2023. Concept learning for interpretable multi-agent reinforcement learning. In *Conference on Robot Learning*, pages 1828–1837. PMLR.