ACTION-GUIDED IN-CONTEXT EXAMPLE RETRIEVAL FOR EMBODIED TASK PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Given only a small set of annotated instruction-plan demonstrations, large language models (LLMs) have shown promising performance in embodied task planning through in-context learning. However, existing approaches primarily retrieve in-context examples based on linguistic or visual similarity, so the retrieved examples may fail to include the actions required by the query task and thus provide limited guidance for generating an appropriate task plan. To address this limitation, we propose an action-guided retrieval approach that retrieves examples relevant to the actions implied by the query to better guide task plan generation. To estimate this relevance, we perform action-guided representation learning based on action sets and action sequences derived from annotated demonstrations, yielding instruction embeddings that capture the action types and their action patterns necessary to complete the task. We further introduce an action-preserving augmentation strategy that diversifies training instructions while preserving their underlying action information to alleviate the scarcity of annotated demonstrations. Experiments on embodied task planning benchmarks show that the proposed method outperforms existing retrieval strategies when integrated with multiple LLMs.

1 INTRODUCTION

Embodied agents are expected to perform tasks specified by natural-language instructions through interactions with their environments. Embodied task planning plays an important role in this process, as it translates an instruction into a sequence of subgoals that describe how the task can be accomplished (Chen et al., 2025; Shin et al., 2025). However, recent studies have identified planning errors as a major source of failure in embodied agents (Yang et al., 2025), and various approaches have been developed to generate accurate task plans (Chen et al., 2025; Kim et al., 2025).

A straightforward way to perform embodied task planning is to train a model that generates task plans from instructions using a large-scale dataset of instruction-plan demonstrations (Kim et al., 2023; Blukis et al., 2022; Wang et al., 2025; Xu et al., 2026). However, obtaining such a dataset requires collecting language instructions across diverse task scenarios and annotating the plans with expert knowledge (Kilinc & Montana, 2022; Sakib et al., 2022; Shridhar et al., 2020). In addition, the annotated plans need to be validated for their executability in physical or simulated environments, making dataset construction substantially time-consuming and labor-intensive (Li et al., 2024; Srivastava et al., 2022). Together, these burdens limit the applicability of task planning methods that rely on large amounts of annotated training data.

To reduce the reliance on extensive annotation efforts, recent studies have explored in-context learning (ICL) with large language models (LLMs) for embodied task planning (Kim et al., 2025; Song et al., 2023). These approaches select a small number of instruction-plan demonstrations and provide them to an LLM as in-context examples for generating a plan for a query instruction. Prior work primarily retrieves examples based on linguistic similarity between the query and training instructions (Song et al., 2023), while a multimodal approach additionally incorporates visual similarity between the agents' initial observations (Kim et al., 2025). By leveraging the pretrained knowledge of LLMs together with retrieved examples, ICL-based methods have shown promising planning performance using only a limited number of training demonstrations.

However, existing retrieval strategies do not explicitly consider action-level relevance, so the retrieved examples may provide LLMs with limited guidance on the action types and their action

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

patterns necessary to accomplish the query task. As shown in Figure 1, our empirical analysis indicates that embodied agents are less likely to complete the task when fewer retrieved examples match the query in terms of the required action types (action-set match) and their action patterns (action-sequence match). This observation highlights the importance of accounting for action-level relevance when retrieving in-context examples for embodied task planning.

To incorporate action-level relevance into in-context example retrieval, we propose **A**ction-**G**uided **R**etrieval (AGR), which retrieves examples based on their relevance to the actions required to accomplish the query task. Since only the query instruction is available at inference time, we perform action-guided representation learning using the training demonstrations to obtain instruction embeddings that capture the action types and their action patterns required to accomplish a task. To facilitate representation learning from limited training demonstrations, we also introduce an action-preserving augmentation strategy to diversify the training instructions while preserving their underlying action information. Using the learned embeddings, AGR retrieves action-relevant examples for the query and provides them to the LLM for task plan generation. We observe that, when integrated with LLMs of different capabilities, AGR enables effective embodied task planning on various benchmarks and outperforms existing ICL-based methods.

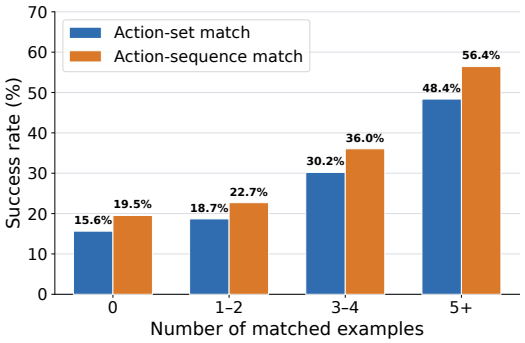


Figure 1: Success rate on the ALFRED validation set for different numbers of action-matched examples among the nine examples retrieved by FLARE (Kim et al., 2025), using GPT-3.5 for task plan generation. A retrieved example is counted as an action-set (action-sequence) match if its action set (action sequence) is identical to that of the query.

We summarize our main contributions as follows:

- We empirically show that embodied agents are more likely to complete a task when more retrieved examples match the query task in terms of the required action types and their action patterns, highlighting the importance of considering action-level relevance when retrieving in-context examples for embodied task planning.
- To retrieve in-context examples based on their action-level relevance to the query, we perform action-guided representation learning to estimate such relevance directly from instructions and employ an action-preserving augmentation strategy to facilitate learning from limited training demonstrations.
- We demonstrate that integrating the action-guided retrieval approach with LLMs of different capabilities substantially improves planning performance over existing retrieval strategies across multiple embodied task planning benchmarks when only a limited number of training demonstrations are available.

2 Related Work

2.1 Embodied Task Planning

Early approaches to embodied task planning typically rely on supervised learning with annotated demonstrations to generate task plans from instructions. FILM (Min et al., 2022) trains a language model to predict task type and task-related arguments from an instruction and constructs a sequence of subgoals using a modular architecture, while HLSM (Blukis et al., 2022) trains a high-level controller to predict subgoals based on a persistent semantic map that accumulates spatial and object information from past observations. CAPEAM (Kim et al., 2023) introduces context-aware planning to generate subgoals based on task-relevant objects and environment-aware memory that tracks changes in object states and locations caused by previous interactions.

As LLMs have shown strong performance in diverse natural language processing tasks, various studies have leveraged LLMs for embodied task planning. Huang et al. (2022a) use an LLM to generate

a sequence of subgoals in natural language and match the generated subgoals to predefined actions using a text encoder. SayCan (Ahn et al., 2022) selects subgoals from predefined candidates based on relevance scores estimated by an LLM and affordance scores that reflect the executability of each candidate in the environment, while Inner Monologue (Huang et al., 2022b) incorporates environmental feedback into the planning process. For planning in complex environments, SayPlan (Rana et al., 2023) provides an LLM with a hierarchical scene graph of the environment and introduces a scene-graph simulator to verify and refine the generated plans. Embodied planning capabilities have also been enhanced through preference and reinforcement learning using large-scale training data and extensive interactions with the environment (Wang et al., 2025; Xu et al., 2026). Unlike approaches that rely on additional training, LLM-Planner (Song et al., 2023) and FLARE (Kim et al., 2025) leverage the pretrained knowledge of LLMs and in-context examples to enable effective embodied task planning with only a limited number of demonstrations.

2.2 In-Context Example Retrieval

In natural language processing, various studies have explored how to retrieve in-context examples that provide LLMs with useful guidance for solving a query. Liu et al. (2022) propose a retrieval method that selects semantically similar examples based on the distance between candidate examples and the query in the text embedding space. An et al. (2023) point out that directly embedding the original input can make retrieval sensitive to linguistic features irrelevant to the target task, and propose retrieving examples based on the similarity between LLM-generated descriptions of the skills required for the query and those for each candidate example. Lin et al. (2025) further argue that semantic similarity may overlook similarities in the problem-solving process, and propose combining semantic similarity with structural similarity between reasoning graphs that capture intermediate reasoning steps. Similarly, Liu et al. (2025) address the limitation of relying solely on semantic similarity between input texts by training the retriever with labeled demonstrations to capture input-output relationships.

In embodied task planning, various approaches leverage in-context examples to guide LLM-based plan generation. A common strategy is to use manually curated or predefined examples that are shared across queries, without dynamically retrieving examples for each query (Singh et al., 2022; Ahn et al., 2022; Liang et al., 2023; Bhat et al., 2025). In contrast, LLM-Planner (Song et al., 2023) retrieves examples based on linguistic similarity between the query and training instructions, under the assumption that linguistically similar examples can provide more useful information for accomplishing the query task. FLARE (Kim et al., 2025) extends this retrieval criterion by additionally considering visual similarity between the agents' initial observations, aiming to retrieve examples that reflect environmental conditions similar to those encountered in the query task. However, such retrieval approaches may fail to retrieve examples involving the actions required to accomplish the query task, motivating us to explicitly incorporate action-level relevance into example retrieval.

3 Proposed Method

We consider embodied task planning with a small training dataset of instruction-plan demonstrations, denoted by $\mathcal{D}_{\text{ori}} = \{(I_i, P_i)\}_{i=1}^{N}$, where N is the number of training demonstrations, I_i is the i-th instruction, and P_i is its ground-truth task plan. A plan consists of a sequence of subgoals, each represented as a triplet (A, O, R): A is a high-level action (e.g., `Pick` or `Heat`), O is the target object of A, and R is the receptacle of O.

From the action components of P_i, we derive two types of action labels: an action-set label $\mathcal{A}_i^{\text{set}}$ and an action-sequence label $\mathcal{A}_i^{\text{seq}}$. The action-set label contains the unique actions appearing in the plan regardless of their order, whereas the action-sequence label preserves the ordered sequence of actions in the plan. For example, for a plan that consists of `Pick`, `Heat`, `Pick`, and `Put` in order, the corresponding labels are $\mathcal{A}_i^{\text{set}} = \{\texttt{Pick}, \texttt{Heat}, \texttt{Put}\}$ and $\mathcal{A}_i^{\text{seq}} = (\texttt{Pick}, \texttt{Heat}, \texttt{Pick}, \texttt{Put})$.

Given a query instruction I_q, our goal is to retrieve K demonstrations from $\mathcal{D}_{\text{ori}}$ and use them as in-context examples to guide LLM_{plan} in generating a plan that accomplishes the task.

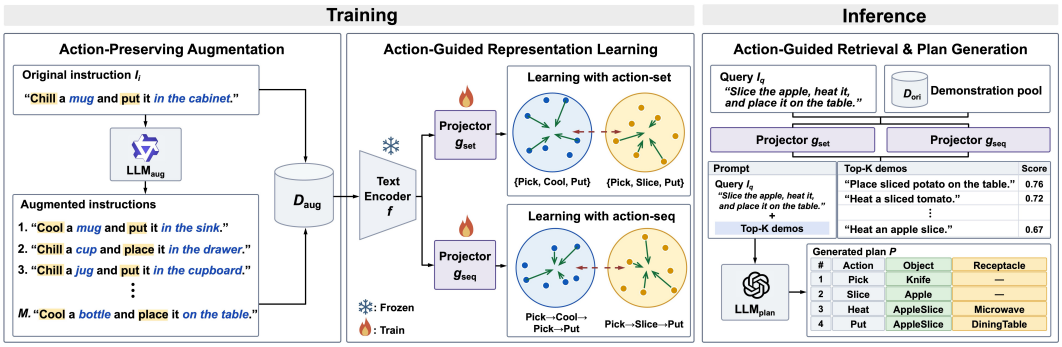


Figure 2: Overview of the action-guided retrieval approach

3.1 Overview

Figure 2 illustrates an overview of AGR. AGR learns action-guided embeddings from action-set and action-sequence labels, capturing action relevance at different levels of granularity and allowing such relevance to be estimated directly from instructions. To facilitate representation learning with limited training demonstrations, we further introduce an action-preserving augmentation strategy that diversifies the training instructions while retaining their action labels. At inference time, AGR combines the similarities from the two representation spaces to retrieve action-relevant demonstrations, which are provided to $\mathrm{LLM}_{\mathrm{plan}}$ as in-context examples for task plan generation. The pseudo-code for AGR is provided in Appendix A.

3.2 Action-Preserving Augmentation

When N is small, the instructions available for learning action-guided representations may exhibit limited diversity. In particular, instructions associated with the same action labels may cover only a narrow range of linguistic expressions, objects, and receptacles. This limited diversity can cause representation learning to capture spurious correlations between particular action labels and specific wording or entities, rather than the action information conveyed by the instructions. To alleviate this potential risk, we propose an action-preserving augmentation strategy that generates diverse variants of each training instruction I_i while maintaining its action labels $\mathcal{A}_i^{\mathrm{set}}$ and $\mathcal{A}_i^{\mathrm{seq}}$.

For each training demonstration (I_i, P_i), we provide both the instruction I_i and its ground-truth task plan P_i to $\mathrm{LLM}_{\mathrm{aug}}$ and generate M alternative instructions $\{\tilde{I}_{i,m}\}_{m=1}^{M}$. Specifically, we prompt $\mathrm{LLM}_{\mathrm{aug}}$ to generate M augmented instructions from the original instruction I_i by varying its linguistic expression and referenced objects or receptacles while preserving the types and order of high-level actions specified by P_i. Each augmented instruction $\tilde{I}_{i,m}$ inherits the action-set and action-sequence labels $\mathcal{A}_i^{\mathrm{set}}$ and $\mathcal{A}_i^{\mathrm{seq}}$ from its source instruction I_i. We then construct the augmented training dataset $\mathcal{D}_{\mathrm{aug}}$ by combining the original and augmented instructions with their corresponding action labels, as shown in Eq. (1).

$$\mathcal{D}_{\mathrm{aug}} = \{(I_i, \mathcal{A}_i^{\mathrm{set}}, \mathcal{A}_i^{\mathrm{seq}})\}_{i=1}^{N} \cup \{(\tilde{I}_{i,m}, \mathcal{A}_i^{\mathrm{set}}, \mathcal{A}_i^{\mathrm{seq}}) \mid 1 \le i \le N,\ 1 \le m \le M\}. \tag{1}$$

The resulting dataset $\mathcal{D}_{\mathrm{aug}}$ is used for action-guided representation learning as described in the following subsection.

3.3 Action-Guided Representation Learning

To capture action-relevant information from instructions, we learn action-guided representation spaces based on the action-set and action-sequence labels in $\mathcal{D}_{\mathrm{aug}}$. We construct a separate representation space for each type of action label using a shared pretrained text encoder and separate projectors. Specifically, each instruction I_i in $\mathcal{D}_{\mathrm{aug}}$ is encoded by the frozen pretrained text encoder $f(\cdot)$ and then mapped through a learnable projector $g_r(\cdot)$, where $r \in \{\mathrm{set}, \mathrm{seq}\}$ denotes the action-label type. The resulting instruction embedding $\mathbf{z}_i^r$ is obtained as shown in Eq. (2).

$$\mathbf{z}_i^r = g_r(f(I_i)). \tag{2}$$

For each instruction I_i, we then define the positive set for action-label type r as $\mathcal{P}_r(i) = \{p \mid p \neq i; \mathcal{A}_p^r = \mathcal{A}_i^r\}$, where instructions sharing the same action label are treated as positives. Based on these positive sets, we define a contrastive loss for each representation space. Specifically, the loss for the instruction I_i in representation space r is defined as shown in Eq. (3).

$$\mathcal{L}_i^r = -\frac{1}{|\mathcal{P}_r(i)|} \sum_{p \in \mathcal{P}_r(i)} \log \frac{\exp\left(\mathrm{sim}\left(\mathbf{z}_i^r, \mathbf{z}_p^r\right)/\tau\right)}{\sum_{j \neq i} \exp\left(\mathrm{sim}\left(\mathbf{z}_i^r, \mathbf{z}_j^r\right)/\tau\right)}, \tag{3}$$

where $\mathrm{sim}(\cdot, \cdot)$ denotes cosine similarity and τ is a temperature parameter. For each action-label type r, the projector g_r is trained by minimizing the average of $\mathcal{L}_i^r$ over all instructions in $\mathcal{D}_{\mathrm{aug}}$. Through this training process, instructions sharing the same action-set label are encouraged to have similar embeddings produced by g_{set}, while instructions sharing the same action-sequence label are encouraged to have similar embeddings produced by g_{seq}. The resulting instruction embeddings enable action-level similarity to be computed directly from a query instruction without access to the ground-truth task plan.

3.4 Action-Guided Retrieval and Plan Generation

At inference time, we use the learned instruction embeddings to retrieve in-context examples that are relevant to the high-level actions implied by the query instruction. Given a query instruction I_q, we obtain two embeddings, $\mathbf{z}_q^{\mathrm{set}}$ and $\mathbf{z}_q^{\mathrm{seq}}$, from g_{set} and g_{seq}, respectively. For each training instruction I_i in $\mathcal{D}_{\mathrm{ori}}$, we then compute a relevance score $s(I_q, I_i)$ by combining the similarities obtained from the two representation spaces as shown in Eq. (4).

$$s(I_q, I_i) = \lambda\, \mathrm{sim}(\mathbf{z}_q^{\mathrm{set}}, \mathbf{z}_i^{\mathrm{set}}) + (1-\lambda)\, \mathrm{sim}(\mathbf{z}_q^{\mathrm{seq}}, \mathbf{z}_i^{\mathrm{seq}}), \tag{4}$$

where $\lambda \in [0, 1]$ balances the two similarity terms, and we set $\lambda = 0.5$ to assign equal weight to each term without tuning. By combining the two similarities, AGR measures the action relevance between the query and each training instruction based on the required action types and their action patterns. The relevance score assigns the highest scores to training instructions that align with the query in both action types and action patterns, so that such instructions are retrieved first. Among the other instructions, those that share relevant action types with the query tend to receive higher scores, even when their action patterns do not align.

We rank the original training demonstrations based on the relevance score and select the top-K demonstrations as in-context examples. Retrieval is restricted to the original demonstrations in $\mathcal{D}_{\mathrm{ori}}$, for which instruction-plan pairs are manually validated by human annotators. The augmented instructions are excluded from retrieval because the objects and receptacles specified in these instructions may be inconsistent with the original task plans. Finally, the selected instruction-plan pairs are provided to $\mathrm{LLM}_{\mathrm{plan}}$ together with the query instruction I_q to generate its task plan.

4 Experiments

4.1 Experimental Setup

Benchmarks. We evaluate our method on three embodied task planning benchmarks: ALFRED (Shridhar et al., 2020), EB-ALFRED (Yang et al., 2025), and WAH-NL (Choi et al., 2024).

ALFRED requires completing household tasks based on natural language instructions in the AI2-THOR simulator and consists of training, validation, and test sets. Both the validation and test sets are divided into seen and unseen scenarios, where seen scenarios represent environments included in the training set, and unseen scenarios represent novel environments absent from the training set. The training set contains 21,023 demonstrations, while the validation set contains 820 seen and 821 unseen tasks, and the test set contains 1,533 seen and 1,529 unseen tasks. Following the few-shot settings of previous studies (Song et al., 2023; Kim et al., 2025), we use the 100 demonstrations adopted in these studies for representation learning and in-context example retrieval.

EB-ALFRED is constructed from the ALFRED validation-seen set, with instructions modified to evaluate different reasoning capabilities involved in embodied task planning. It contains 300 evaluation tasks organized into six subsets, with 50 tasks in each subset. Since EB-ALFRED provides only an evaluation set, we use the 100 demonstrations selected from the ALFRED training set for representation learning and in-context example retrieval.

Table 1: Comparison on the ALFRED test set under few-shot and full-shot settings. We excerpt the SR and GC results of the comparison methods from Kim et al. (2025). For ICL-based methods using the same LLM, the best results are highlighted in **bold**.

Setting	Type	LLM	Method	ALFRED-seen		ALFRED-unseen	
				SR	GC	SR	GC
Few	Sup.	-	HLSM	0.82	6.88	0.61	3.72
			FILM	0.00	4.23	0.20	6.71
			CAPEAM	0.00	3.90	0.20	6.63
	ICL	text-davinci-003	LLM-Planner	18.20	26.77	16.42	23.37
		LLaMA2	FLARE	16.96	24.84	17.79	27.40
			AGR	**28.71**	**42.83**	**22.44**	**36.37**
		GPT-3.5	FLARE	32.55	42.02	31.79	43.94
			AGR	**33.49**	**44.10**	**35.38**	**47.74**
		GPT-4	FLARE	40.05	48.84	40.88	51.72
		GPT-4o	AGR	39.60	53.39	41.20	49.29
Full	Sup.	-	HLSM	29.94	41.21	20.27	30.31
			FILM	28.83	39.55	27.80	38.52
			CAPEAM	51.79	60.50	46.11	57.33

WAH-NL consists of 250 training tasks and 100 test tasks based on the Watch-And-Help (WAH) dataset, with natural language instructions provided for embodied task planning. The 250 training tasks are represented by 416 instructions and the 100 test tasks by 195 instructions; the former are used for representation learning and in-context example retrieval, and the latter are used for evaluation. The tasks are executed in the VirtualHome simulator.

Evaluation Metrics. To evaluate whether the generated task plans lead to successful task completion, we report success rate (SR) and goal-condition success rate (GC). SR measures the percentage of successfully completed tasks, while GC measures the percentage of satisfied goal conditions, accounting for partial task completion. We also report high-level planning accuracy (HLP) to directly evaluate the generated task plans. A generated plan is considered correct only if it exactly matches the ground-truth task plan. Since such plans are unavailable for both the ALFRED test set and the WAH-NL test tasks, we report HLP only on the ALFRED validation set and EB-ALFRED.

Implementation. We use a pretrained BERT-base as the text encoder f and train separate projectors g_{set} and g_{seq}. Each projector consists of a single linear layer that maps 768-dimensional embeddings from BERT to 128 dimensions. The projectors are trained for 100 epochs using AdamW with a learning rate of 0.01, and the temperature parameter τ in the contrastive loss is set to 0.1. For data augmentation, we use Qwen2.5-7B-Instruct as LLM_{aug} and generate 20 augmented instructions for each training demonstration. The augmented instructions are used only to train the projectors, while in-context example retrieval is performed over the original instructions in $\mathcal{D}_{\text{ori}}$. For task plan generation, we use LLaMA-2-13B, GPT-3.5 (gpt-3.5-turbo-instruct), and GPT-4o (gpt-4o-2024-08-06) as LLM_{plan}. We set the sampling temperature of all LLMs to zero for reproducibility. At inference time, we retrieve nine in-context examples, following prior embodied task planning approaches under the few-shot setting for a fair comparison (Song et al., 2023; Kim et al., 2025).

For ALFRED and EB-ALFRED, where executing a generated task plan requires a separate navigation and object-interaction procedure in AI2-THOR, we follow the execution procedure used in FLARE (Kim et al., 2025), except that AGR does not perform replanning. For WAH-NL, we directly execute the generated task plans in VirtualHome, where navigation and interaction with objects are handled by the simulator itself.

Comparison Methods. We compare AGR with LLM-Planner (Song et al., 2023) and FLARE (Kim et al., 2025), which focus on embodied task planning with limited demonstrations. Both LLM-Planner and FLARE incorporate replanning during task execution, whereas AGR executes the initially generated task plan without replanning. We also include three supervised approaches: HLSM (Blukis et al., 2022), FILM (Min et al., 2022), and CAPEAM (Kim et al., 2023). Since these supervised approaches were developed specifically for ALFRED, we report their results only on ALFRED and EB-ALFRED. Unless otherwise stated, all reported results for the comparison methods are obtained from our own experiments.

Table 2: Comparison on the ALFRED validation set and EB-ALFRED under the few-shot setting, including supervised and ICL-based methods. For ICL-based methods using the same LLM, the best results are highlighted in **bold**.

Type	LLM	Method	ALFRED-seen			ALFRED-unseen			EB-ALFRED		
			SR	GC	HLP	SR	GC	HLP	SR	GC	HLP
Sup.	–	HLSM	0.12	6.02	0.00	0.00	7.03	0.00	0.00	4.03	0.00
		FILM	1.46	9.00	0.00	1.83	10.28	0.00	0.33	6.45	0.33
		CAPEAM	2.07	11.13	0.00	1.46	11.77	0.00	3.33	10.42	0.00
ICL	LLaMA2	LLM-Planner	6.10	16.26	8.66	4.38	14.86	9.74	3.00	13.65	3.67
		FLARE	**24.88**	37.08	33.54	22.90	36.32	35.93	21.67	35.89	23.67
		AGR	23.78	**37.60**	**35.73**	**26.19**	**40.00**	**44.09**	**23.67**	**40.83**	**30.33**
	GPT-3.5	LLM-Planner	15.37	25.60	27.07	16.93	30.94	31.91	7.67	17.30	14.00
		FLARE	27.56	38.64	41.46	28.38	39.67	50.79	29.67	40.83	30.00
		AGR	**34.51**	**45.09**	**50.61**	**34.96**	**47.17**	**59.56**	**33.00**	**44.60**	**37.33**
	GPT-4o	LLM-Planner	31.22	41.35	50.85	36.42	49.86	59.32	16.67	28.09	28.67
		FLARE	38.29	47.89	59.76	34.84	46.13	67.36	**37.67**	47.33	45.00
		AGR	**40.37**	**49.50**	**61.83**	**38.25**	**50.28**	**68.94**	**37.67**	**48.89**	**46.33**

4.2 Main Results

ALFRED Test Set. To evaluate task planning performance under the few-shot setting, Table 1 compares AGR with supervised and ICL-based approaches on the ALFRED test set. For the supervised approaches, we report results under both few-shot and full-shot settings. The results of the proposed method are obtained from the ALFRED leaderboard, while the results of the comparison methods are excerpted from Kim et al. (2025).

Supervised approaches such as HLSM, FILM, and CAPEAM perform well in the full-shot setting, whereas their performance substantially degrades when only 100 demonstrations are available for training. In contrast, AGR shows a substantial performance margin over these supervised approaches in the few-shot setting and, with GPT-3.5 and GPT-4o, even outperforms full-shot HLSM and FILM across both SR and GC in seen and unseen environments. Among the ICL-based methods, AGR achieves effective task planning performance across different LLMs. While LLM-Planner uses a different LLM and is therefore not directly comparable, AGR outperforms FLARE in all cases when using the same LLMs, LLaMA2 and GPT-3.5. For the GPT-4 family, FLARE is evaluated with GPT-4, whereas AGR is evaluated with GPT-4o because the GPT-4 version used in FLARE is no longer available and other GPT-4 variants are deprecated. Under this setting, AGR achieves higher SR in the unseen environment and higher GC in the seen environment. These results suggest that retrieving examples based on the actions implied by the instruction helps the LLM generate task plans that are more likely to lead to successful task execution and satisfy more goal conditions.

ALFRED Validation Set and EB-ALFRED. Table 2 compares AGR with the baselines on the ALFRED validation set and EB-ALFRED under the few-shot setting. For the ICL-based methods, we evaluate LLM-Planner, FLARE, and AGR using the same LLMs so that differences in LLM capability do not affect the comparison of task planning performance. In addition to SR and GC, we report HLP to directly evaluate the accuracy of the generated task plans.

Similar to the results in Table 1, the supervised approaches show considerably lower performance than the ICL-based methods in the few-shot setting. These results again demonstrate the limited applicability of supervised approaches in embodied task planning when only a small number of demonstrations are available. For the ICL-based methods, AGR achieves higher HLP in all cases, suggesting that the proposed retrieval approach more effectively guides the LLM to generate task plans that exactly match the ground-truth plans. This advantage is also reflected in higher SR and GC achieved by AGR across seen and unseen environments in most cases. On ALFRED-seen with LLaMA2, FLARE achieves slightly higher SR, while AGR achieves higher GC and HLP in the same setting. Notably, the proposed method also remains effective on EB-ALFRED, where the original ALFRED instructions are modified to require diverse reasoning capabilities, such as common-sense reasoning and complex instruction understanding. Detailed results for the seven ALFRED task types and six EB-ALFRED subsets are provided in Appendix B.

WAH-NL. To assess the effectiveness of AGR beyond ALFRED and EB-ALFRED, both built on the AI2-THOR simulator, we evaluate the proposed method on WAH-NL, which is built on

Table 4: Effect of the projectors used for action-guided retrieval. g_{set} and g_{seq} denote the projectors trained with action-set and action-sequence supervision, respectively. For task plan generation, we use GPT-3.5 in all experiments. The best results are highlighted in **bold**.

g_{set}	g_{seq}	ALFRED-seen		ALFRED-unseen		EB-ALFRED		WAH-NL	
		SR	GC	SR	GC	SR	GC	SR	GC
✔	✔	**34.51**	**45.09**	34.96	47.17	**33.00**	44.60	**8.21**	**40.67**
✔	✘	33.78	44.81	**35.32**	**47.26**	31.00	**44.73**	7.18	39.12
✘	✔	33.78	44.14	33.01	44.48	30.67	43.56	7.72	40.40

the VirtualHome simulator. Compared with the ALFRED-based benchmarks, WAH-NL involves a different environment and requires different action types and patterns to complete tasks.

Table 3 compares AGR with LLM-Planner and FLARE on WAH-NL. WAH-NL tasks often require completing multiple subgoals through repeated interactions with various objects (Choi et al., 2024), making complete task success challenging and resulting in relatively low SR across the compared methods. Nevertheless, AGR consistently achieves the highest SR and GC across the evaluated LLMs. Compared with FLARE, AGR improves SR by 2.05% to 4.10% and GC by 1.64% to 3.02%. The consistent performance gains on WAH-NL suggest that the proposed action-guided retrieval approach is not limited to a specific benchmark or simulator and can remain effective across different embodied task planning settings.

Table 3: Comparison of ICL-based methods on the WAH-NL benchmark. For methods using the same LLM, the best results are highlighted in **bold**.

LLM	Method	WAH-NL	
		SR	GC
LLaMA2	LLM-Planner	3.08	25.42
	FLARE	3.59	37.50
	AGR	**7.69**	**40.52**
GPT-3.5	LLM-Planner	3.08	26.38
	FLARE	5.13	38.45
	AGR	**8.21**	**40.67**
GPT-4o	LLM-Planner	5.64	37.93
	FLARE	7.69	40.40
	AGR	**9.74**	**42.04**

In Appendix C, we further compare the ICL-based methods on the ALFRED validation set, EB-ALFRED, and WAH-NL without replanning, and observe that AGR achieves the highest SR and GC across all evaluated benchmarks.

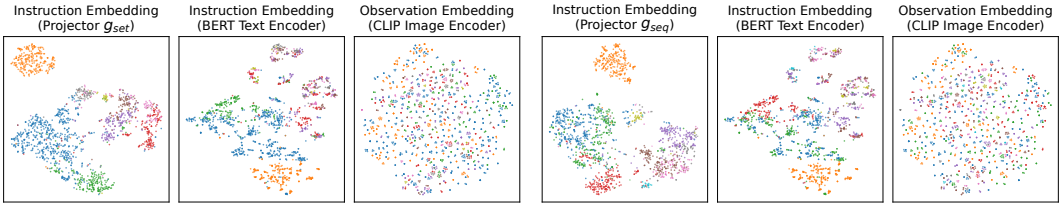
4.3 Analysis of Action-Guided Retrieval

Ablation Study. We examine the effects of the two projectors used for in-context example retrieval and the number of augmented instructions used for representation learning. Table 4 compares task planning performance when using g_{set} and g_{seq} individually and when using both projectors together for in-context example retrieval. Using both projectors achieves the best or second-best SR and GC across all benchmarks, showing consistent and effective task planning performance. When used individually, g_{set} achieves SR and GC comparable to or better than those of g_{seq} on the ALFRED-based benchmarks. In contrast, g_{seq} achieves higher SR and GC than g_{set} on WAH-NL, which requires more interactions to complete tasks than ALFRED. We further analyze how each projector affects the retrieved examples in Appendix D.

Table 5 examines the effect of the number of augmented instructions M used for representation learning. Compared with $M = 0$, using 10 augmented instructions substantially improves both SR and GC across all benchmarks, suggesting that increasing instruction diversity through our augmentation strategy is beneficial for learning action-guided representations when only a limited number of demonstrations are available. However, using more augmented instructions does not necessarily lead to better task planning performance. While $M = 20$ achieves the best performance on ALFRED-seen and EB-ALFRED, $M = 30$ performs best on ALFRED-unseen and WAH-NL. We also qualitatively examined the augmented instructions. Across the evaluated values of M, the underlying action information of the original instructions was effectively preserved. However, increasing M caused redundancy among the augmented instructions generated from the same original instruction. We observed no redundancy when $M = 10$, whereas approximately 0.8% and 3.2% of the augmented instructions were duplicates when $M = 20$ and $M = 30$, respectively. A larger number of duplicates may bias action-guided representation learning by overemphasizing redundant

Table 5: Effect of the number of augmented instructions M used for representation learning. For task plan generation, we use GPT-3.5 in all experiments. The best results are highlighted in **bold**.

M	ALFRED-seen		ALFRED-unseen		EB-ALFRED		WAH-NL	
	SR	GC	SR	GC	SR	GC	SR	GC
0	29.63	40.16	30.82	43.30	26.33	38.23	7.21	39.20
10	32.80	43.86	34.47	46.27	31.67	43.82	8.21	42.29
20	**34.51**	**45.09**	34.96	47.17	**33.00**	**44.60**	8.21	40.67
30	33.17	44.05	**35.57**	**47.45**	30.33	43.95	**9.23**	**43.40**



(a) Embeddings colored by action sets

(b) Embeddings colored by action sequences

Figure 3: Embeddings for samples from the ALFRED validation set, colored by (a) action-set labels and (b) action-sequence labels. We compare the instruction embeddings produced by g_{set} and g_{seq} with instruction embeddings obtained from a pretrained BERT text encoder and observation embeddings obtained from a pretrained CLIP image encoder using the agent's initial observation.

instructions. Considering that increasing M beyond 20 does not provide consistent performance gains while introducing greater redundancy, we set $M = 20$ for our main experiments.

Analysis of Learned Embeddings. For in-context example retrieval in embodied task planning, existing approaches typically rely on instruction embeddings obtained from a pretrained BERT text encoder and, in some cases, observation embeddings obtained from a pretrained CLIP image encoder. To compare these embeddings with the instruction embeddings learned by AGR, we visualize each embedding space in Figure 3.

As shown in Figure 3, the instruction embeddings obtained from BERT exhibit several local clusters. However, samples with different action labels often appear within the same cluster. The observation embeddings obtained from CLIP show even weaker separation among action labels. In contrast, the instruction embeddings produced by g_{set} and g_{seq} form clearer clusters, with samples sharing the same action-set or action-sequence label grouped more closely. This clearer separation allows AGR to retrieve in-context examples whose required actions are relevant to the query instruction. In addition, we observe that AGR effectively retrieves examples that match the query in terms of both action set and action sequence, as shown in Appendix D.

5 Conclusion

In this paper, we introduced an action-guided retrieval approach to better guide LLMs in generating appropriate embodied task plans from limited training demonstrations. Our empirical analysis showed that task success tends to improve as more retrieved examples match the query in terms of the required action types and their action patterns, highlighting the importance of considering action-level relevance in in-context example retrieval. To capture such relevance directly from instructions, AGR performs action-guided representation learning and retrieves in-context examples based on the learned instruction embeddings. We also introduced an action-preserving augmentation strategy that increases instruction diversity while preserving the underlying action information, thereby supporting representation learning from limited training demonstrations. Experimental results across multiple embodied task planning benchmarks demonstrate that the proposed method outperforms existing retrieval strategies.

Despite these results, AGR has some limitations. First, it relies on ground-truth plans to obtain action-set and action-sequence labels for representation learning. Second, our evaluation is limited to simulated environments. Future work could explore learning action-guided representations without plan annotations and extending AGR to embodied task planning in the physical world.

AI USE STATEMENT

In this work, we used Qwen2.5-7B-Instruct to generate augmented instructions for representation learning, as described in the paper. We also used generative AI tools to assist with manuscript polishing and formatting, as well as with writing Python code for visualization. All AI-assisted text, code, and generated data were reviewed and verified by the authors. We take responsibility for the final content of this work, including text, code, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses existing embodied task planning benchmarks and does not involve the collection of human-subject data. All experiments are conducted in simulated environments. Although the proposed method is evaluated only for research purposes in simulation, applying embodied task planning systems to the physical world may require additional consideration of safety and reliability.

REPRODUCIBILITY STATEMENT

To ensure reproducibility, we provide detailed descriptions of the proposed method in Section 3 and the benchmark settings, evaluation metrics, and implementation details in Section 4. We also provide the source code for the proposed method as supplementary material (`AGR_codes.zip`).

REFERENCES

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as I can, not as I say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Shengnan An, Bo Zhou, Zeqi Lin, Qiang Fu, Bei Chen, Nanning Zheng, Weizhu Chen, and Jian-Guang Lou. Skill-based few-shot selection for in-context learning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 13472–13492, 2023.

Vineet Bhat, Ali Umut Kaypak, Prashanth Krishnamurthy, Ramesh Karri, and Farshad Khorrami. Grounding large language models for robot task planning using closed-loop state feedback. *Advanced Robotics Research*, pp. e202500072, 2025.

Valts Blukis, Chris Paxton, Dieter Fox, Animesh Garg, and Yoav Artzi. A persistent spatial semantic representation for high-level natural language instruction execution. In *Proceedings of the 5th Conference on Robot Learning*, pp. 706–717. PMLR, 2022.

Yifan Chen, Mingjie Wei, Xuesong Wang, Yuanxing Liu, Jizhe Wang, Hao Song, Longxuan Ma, Donglin Di, Churui Sun, Kaifeng Liu, et al. Embodied AI: A survey on the evolution from perceptive to behavioral intelligence. *SmartBot*, 1(3):e70003, 2025.

Jae-Woo Choi, Youngwoo Yoon, Hyobin Ong, Jaehong Kim, and Minsu Jang. LoTa-Bench: Benchmarking language-oriented task planners for embodied agents. In *International Conference on Learning Representations (ICLR)*, 2024.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 9118–9147. PMLR, 2022a.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, et al. Inner Monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022b.

Ozsel Kilinc and Giovanni Montana. Reinforcement learning for robotic manipulation using simulated locomotion demonstrations. *Machine Learning*, 111(2):465–486, 2022.

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

Taewoong Kim, Byeonghwi Kim, and Jonghyun Choi. Multi-modal grounded planning and efficient replanning for learning embodied agents with a few examples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 4329–4337, 2025.

Kanxue Li, Baosheng Yu, Qi Zheng, Yibing Zhan, Yuhui Zhang, Tianle Zhang, Yijun Yang, Yue Chen, Lei Sun, Qiong Cao, et al. MuEP: A multimodal benchmark for embodied planning with foundation models. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pp. 129–138, 2024.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9493–9500. IEEE, 2023.

Yukang Lin, Bingchen Zhong, Shuoran Jiang, Joanna Siebert, and Qingcai Chen. Reasoning graph enhanced exemplars retrieval for in-context learning. In *Proceedings of the 31st International Conference on Computational Linguistics*, pp. 9737–9759, 2025.

Hui Liu, Wenya Wang, Hao Sun, Chris Xing Tian, Chenqi Kong, Xin Dong, and Haoliang Li. Unraveling the mechanics of learning-based demonstration selection for in-context learning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2623–2641, 2025.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, William B Dolan, Lawrence Carin, and Weizhu Chen. What makes good in-context examples for GPT-3? In *Proceedings of Deep Learning Inside Out (DeeLIO 2022): The 3rd Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pp. 100–114, 2022.

So Yeon Min, Devendra Singh Chaplot, Pradeep Ravikumar, Yonatan Bisk, and Ruslan Salakhutdinov. FILM: Following instructions in language with modular methods. In *International Conference on Learning Representations (ICLR)*, 2022.

Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. SayPlan: Grounding large language models using 3D scene graphs for scalable robot task planning. *arXiv preprint arXiv:2307.06135*, 2023.

Md Sadman Sakib, David Paulius, and Yu Sun. Approximate task tree retrieval in a knowledge network for robotic cooking. *IEEE Robotics and Automation Letters*, 7(4):11492–11499, 2022.

Suyeon Shin, Sujin Jeon, Junghyun Kim, Gi-Cheon Kang, and Byoung-Tak Zhang. Socratic Planner: Self-QA-based zero-shot planning for embodied instruction following. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16217–16224. IEEE, 2025.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10740–10749, 2020.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. ProgPrompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

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

Sanjana Srivastava, Chengshu Li, Michael Lingelbach, Roberto Martín-Martín, Fei Xia, Kent Elliott Vainio, Zheng Lian, Cem Gokmen, Shyamal Buch, Karen Liu, et al. BEHAVIOR: Benchmark for everyday household activities in virtual, interactive, and ecological environments. In *Proceedings of the 5th Conference on Robot Learning*, pp. 477–490. PMLR, 2022.

Siyin Wang, Zhaoye Fei, Qinyuan Cheng, Shiduo Zhang, Panpan Cai, Jinlan Fu, and Xipeng Qiu. World modeling makes a better planner: Dual preference optimization for embodied task planning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 21518–21537, 2025.

Peiran Xu, Jiaqi Zheng, and Yadong Mu. RoboAgent: Chaining basic capabilities for embodied task planning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15276–15290, 2026.

Rui Yang, Hanyang Chen, Junyu Zhang, Mark Zhao, Cheng Qian, Kangrui Wang, Qineng Wang, Teja Venkat Koripella, Marziyeh Movahedi, Manling Li, et al. EmbodiedBench: Comprehensive benchmarking multi-modal large language models for vision-driven embodied agents. *arXiv preprint arXiv:2502.09560*, 2025.

A PSEUDO-CODE

Algorithm A1 summarizes the overall procedure of AGR. During training, AGR performs action-preserving augmentation and learns action-guided representations using action-set and action-sequence labels. At inference time, the learned representations are used to retrieve action-relevant in-context examples, which are then provided to $\mathrm{LLM}_{\mathrm{plan}}$ for task plan generation.

Algorithm A1 Action-Guided Retrieval (AGR)

Input: Training demonstrations $\mathcal{D}_{\mathrm{ori}} = \{(I_i, P_i)\}_{i=1}^{N}$, query instruction I_q, augmentation size M, number of in-context examples K
Output: Task plan for I_q

 # Action-Preserving Augmentation
for each $(I_i, P_i) \in \mathcal{D}_{\mathrm{ori}}$ **do**
    Derive $\mathcal{A}_i^{\mathrm{set}}$ and $\mathcal{A}_i^{\mathrm{seq}}$ from P_i
    Generate M augmented instructions $\{\tilde{I}_{i,m}\}_{m=1}^{M}$ with $\mathrm{LLM}_{\mathrm{aug}}$ using I_i and P_i
    Assign $\mathcal{A}_i^{\mathrm{set}}$ and $\mathcal{A}_i^{\mathrm{seq}}$ to the augmented instructions
end for
Construct $\mathcal{D}_{\mathrm{aug}}$ using Eq. (1)
 # Action-Guided Representation Learning
for $r \in \{\mathrm{set}, \mathrm{seq}\}$ **do**
    Obtain $\mathbf{z}_i^r$ for all instructions in $\mathcal{D}_{\mathrm{aug}}$ using Eq. (2)
    Train g_r using action labels $\mathcal{A}_i^r$ by minimizing Eq. (3)
end for
 # Action-Guided Retrieval
Obtain $\mathbf{z}_q^{\mathrm{set}}$ and $\mathbf{z}_q^{\mathrm{seq}}$ from I_q using Eq. (2)
for each $(I_i, P_i) \in \mathcal{D}_{\mathrm{ori}}$ **do**
    Obtain $\mathbf{z}_i^{\mathrm{set}}$ and $\mathbf{z}_i^{\mathrm{seq}}$ using Eq. (2)
    Compute $s(I_q, I_i)$ using Eq. (4)
end for
Retrieve the top-K examples based on $s(I_q, I_i)$
 # Task Plan Generation
Generate the task plan for I_q using $\mathrm{LLM}_{\mathrm{plan}}$ with the K retrieved examples

B DETAILED RESULTS ON ALFRED VALIDATION AND EB-ALFRED

We provide detailed results for ALFRED and EB-ALFRED. Tables B1 and B2 report performance across the seven ALFRED task types in seen and unseen environments, respectively, while Table B3 reports performance across the six EB-ALFRED subsets. Following the official ALFRED evaluation protocol (Shridhar et al., 2020), the Overall column reports performance computed over all episodes.

Across both seen and unseen environments in ALFRED, the task-type breakdown shows that AGR improves embodied task planning performance across multiple task types. For SR and GC, AGR shows improvements across several task types, particularly Clean and Heat. For HLP, the advantage of AGR is observed across a larger number of task types. With GPT-3.5, AGR achieves the best performance on six of the seven task types in both environments. With GPT-4o, AGR achieves the best performance on six task types in seen environments and five task types in unseen environments. On Clean and Heat, AGR achieves the best HLP performance with all three LLMs in both environments.

On EB-ALFRED, AGR also improves performance across multiple subsets. For SR, AGR achieves the best performance on five of the six subsets with both LLaMA2 and GPT-3.5. For GC, AGR achieves the best performance on five, four, and four subsets with LLaMA2, GPT-3.5, and GPT-4o, respectively. For HLP, the advantage of AGR is observed across more subsets. AGR achieves the best performance on all six subsets with LLaMA2 and on five of the six subsets with GPT-3.5, while achieving the best performance on five subsets with GPT-4o.

Table B1: Comparison of supervised and ICL-based methods by task type on the ALFRED validation set in seen environments under the few-shot setting. The Overall column reports performance computed over all episodes in the seen scenarios of the validation set. For ICL-based methods using the same LLM, the best result in each column is highlighted in **bold**.

(a) Success rate (SR)

Type	LLM	Method	Examine	Pick	Stack	Clean	Cool	Heat	Pick Two	Overall
Sup.	–	HLSM	0.00	0.70	0.00	0.00	0.00	0.00	0.00	0.12
		FILM	1.06	0.00	3.48	0.00	0.79	5.61	0.00	1.46
		CAPEAM	1.06	0.00	5.22	0.00	0.00	9.35	0.00	2.07
ICL	LLaMA2	LLM-Planner	13.83	8.45	**6.09**	6.25	0.79	0.00	8.06	6.10
		FLARE	46.81	**41.55**	**6.09**	27.68	**11.11**	19.63	**22.58**	**24.88**
		AGR	**47.87**	32.39	3.48	**31.25**	9.52	**23.36**	**22.58**	23.78
	GPT-3.5	LLM-Planner	31.91	26.06	10.43	16.07	3.17	6.54	14.52	15.37
		FLARE	**50.00**	46.48	13.04	11.61	**19.05**	24.30	28.23	27.56
		AGR	45.74	**52.11**	**14.78**	**44.64**	16.67	**32.71**	**34.68**	**34.51**
	GPT-4o	LLM-Planner	42.55	38.73	**15.65**	33.04	26.19	46.73	18.55	31.22
		FLARE	**56.38**	52.82	13.04	36.61	**27.78**	**50.47**	33.06	38.29
		AGR	54.26	**54.93**	11.30	**50.89**	**27.78**	48.60	**36.29**	**40.37**

(b) Goal-condition success rate (GC)

Type	LLM	Method	Examine	Pick	Stack	Clean	Cool	Heat	Pick Two	Overall
Sup.	–	HLSM	1.60	0.70	1.56	12.03	9.35	8.97	1.15	6.02
		FILM	0.53	1.32	5.18	13.91	12.41	19.22	1.16	9.00
		CAPEAM	0.53	1.32	6.81	14.78	12.65	29.25	1.16	11.13
ICL	LLaMA2	LLM-Planner	30.85	8.45	13.51	18.62	14.87	14.67	15.38	16.26
		FLARE	**61.70**	**41.55**	**22.34**	42.41	**31.18**	39.67	**37.31**	37.08
		AGR	57.98	32.39	21.56	**49.86**	28.78	**47.01**	33.85	**37.60**
	GPT-3.5	LLM-Planner	55.85	26.06	18.44	31.23	18.47	19.57	26.54	25.60
		FLARE	**64.89**	46.48	**28.57**	26.36	**35.25**	47.01	40.38	38.64
		AGR	56.38	**52.11**	28.31	**60.17**	**35.25**	**50.27**	**46.15**	**45.09**
	GPT-4o	LLM-Planner	59.04	38.73	21.56	46.70	39.81	56.52	33.08	41.35
		FLARE	**68.09**	52.82	**28.31**	48.71	**44.12**	**61.14**	45.77	47.89
		AGR	62.77	**54.93**	26.49	**63.61**	42.45	60.33	**48.08**	**49.50**

(c) High-level planning accuracy (HLP)

Type	LLM	Method	Examine	Pick	Stack	Clean	Cool	Heat	Pick Two	Overall
Sup.	–	HLSM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		FILM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		CAPEAM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ICL	LLaMA2	LLM-Planner	17.02	14.79	4.35	7.14	2.38	0.93	13.71	8.66
		FLARE	57.45	**42.25**	**16.52**	30.36	**21.43**	19.63	48.39	33.54
		AGR	**62.77**	37.32	12.17	**33.93**	19.05	**40.19**	**50.00**	**35.73**
	GPT-3.5	LLM-Planner	52.13	42.25	13.91	24.11	10.32	6.54	40.32	27.07
		FLARE	62.77	55.63	**26.09**	27.68	26.98	24.30	65.32	41.46
		AGR	**69.15**	**66.20**	23.48	**37.50**	**37.30**	**46.73**	**72.58**	**50.61**
	GPT-4o	LLM-Planner	65.96	56.34	25.22	50.00	53.17	54.21	52.42	50.85
		FLARE	77.66	68.31	**34.78**	56.25	55.56	50.47	**75.00**	59.76
		AGR	**79.79**	**70.42**	31.30	**58.04**	**57.94**	**60.75**	**75.00**	**61.83**

Table B2: Comparison of supervised and ICL-based methods by task type on the ALFRED validation set in unseen environments under the few-shot setting. The Overall column reports performance computed over all episodes in the unseen scenarios of the validation set. For ICL-based methods using the same LLM, the best result in each column is highlighted in **bold**.

(a) Success rate (SR)

Type	LLM	Method	Examine	Pick	Stack	Clean	Cool	Heat	Pick Two	Overall
Sup.	–	HLSM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		FILM	0.58	0.00	1.83	0.00	3.67	5.88	0.00	1.83
		CAPEAM	0.58	0.00	2.75	0.00	1.83	4.41	0.00	1.46
ICL	LLaMA2	LLM-Planner	15.61	2.00	0.00	2.65	1.83	1.47	0.00	4.38
		FLARE	**50.87**	**21.00**	8.26	15.04	3.67	24.26	**19.75**	22.90
		AGR	49.13	11.00	**9.17**	**31.86**	**11.01**	**36.03**	14.81	**26.19**
	GPT-3.5	LLM-Planner	27.75	16.00	16.51	21.24	9.17	5.88	18.52	16.93
		FLARE	58.38	20.00	**18.35**	10.62	10.09	27.21	**39.51**	28.38
		AGR	**66.47**	**22.00**	16.51	**28.32**	**11.01**	**45.59**	32.10	**34.96**
	GPT-4o	LLM-Planner	40.46	**25.00**	**22.02**	**38.05**	**50.46**	48.53	19.75	36.42
		FLARE	62.43	23.00	20.18	19.47	18.35	45.59	**35.80**	34.84
		AGR	**65.90**	21.00	**22.02**	37.17	15.60	**51.47**	32.10	**38.25**

(b) Goal-condition success rate (GC)

Type	LLM	Method	Examine	Pick	Stack	Clean	Cool	Heat	Pick Two	Overall
Sup.	–	HLSM	3.76	0.00	0.00	14.37	11.58	9.93	0.00	7.03
		FILM	0.29	0.00	5.11	13.58	14.20	22.02	5.24	10.28
		CAPEAM	0.29	0.00	6.07	13.58	14.78	26.29	9.42	11.77
ICL	LLaMA2	LLM-Planner	34.68	2.00	2.45	16.38	18.08	12.36	4.17	14.86
		FLARE	**68.79**	**21.00**	**15.90**	35.06	22.60	42.60	**33.33**	36.32
		AGR	66.47	11.00	15.29	**47.99**	**27.40**	**53.86**	25.52	**40.00**
	GPT-3.5	LLM-Planner	53.18	16.00	21.71	31.61	**29.10**	21.63	38.54	30.94
		FLARE	74.28	20.00	**23.55**	25.57	25.99	44.59	**54.17**	39.67
		AGR	**78.32**	**22.00**	22.94	**46.84**	26.84	**61.59**	49.48	**47.17**
	GPT-4o	LLM-Planner	63.29	**25.00**	**30.58**	47.70	**61.30**	57.84	35.42	49.86
		FLARE	76.30	23.00	25.99	35.92	32.20	59.16	**51.56**	46.13
		AGR	**78.32**	21.00	27.52	**54.31**	29.66	**65.12**	49.48	**50.28**

(c) High-level planning accuracy (HLP)

Type	LLM	Method	Examine	Pick	Stack	Clean	Cool	Heat	Pick Two	Overall
Sup.	–	HLSM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		FILM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		CAPEAM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ICL	LLaMA2	LLM-Planner	21.97	16.00	4.59	4.42	0.92	2.94	13.58	9.74
		FLARE	37.57	**39.00**	**33.03**	39.82	26.61	29.41	**50.62**	35.93
		AGR	**54.91**	38.00	22.02	**60.18**	**33.94**	**46.32**	45.68	**44.09**
	GPT-3.5	LLM-Planner	55.49	42.00	26.61	34.51	14.68	8.82	34.57	31.91
		FLARE	82.66	57.00	**44.04**	38.94	29.36	34.56	56.79	50.79
		AGR	**84.97**	**59.00**	35.78	**61.06**	**50.46**	**53.68**	**58.02**	**59.56**
	GPT-4o	LLM-Planner	68.21	60.00	34.86	65.49	**71.56**	63.24	40.74	59.32
		FLARE	80.35	64.00	**46.79**	72.57	69.72	**66.18**	62.96	67.36
		AGR	**82.08**	**66.00**	43.12	**82.30**	68.81	**66.18**	**65.43**	**68.94**

Table B3: Comparison of supervised and ICL-based methods across the six EB-ALFRED subsets. The Overall column reports performance computed over all episodes in EB-ALFRED. For ICL-based methods using the same LLM, the best result in each column is highlighted in **bold**.

(a) Success rate (SR)

Type	LLM	Method	Base	Common	Complex	Long	Spatial	Visual	Overall
Sup.	–	HLSM	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		FILM	0.00	2.00	0.00	0.00	0.00	0.00	0.33
		CAPEAM	2.00	4.00	2.00	10.00	2.00	0.00	3.33
ICL	LLaMA2	LLM-Planner	2.00	4.00	2.00	2.00	6.00	2.00	3.00
		FLARE	32.00	14.00	24.00	12.00	18.00	**30.00**	21.67
		AGR	**36.00**	**22.00**	**26.00**	**20.00**	**22.00**	16.00	**23.67**
	GPT-3.5	LLM-Planner	8.00	4.00	10.00	0.00	14.00	10.00	7.67
		FLARE	40.00	30.00	**30.00**	14.00	28.00	36.00	29.67
		AGR	**42.00**	**36.00**	26.00	**20.00**	**32.00**	**42.00**	**33.00**
	GPT-4o	LLM-Planner	14.00	10.00	22.00	14.00	22.00	18.00	16.67
		FLARE	42.00	40.00	**42.00**	**24.00**	**40.00**	**38.00**	**37.67**
		AGR	**48.00**	**48.00**	34.00	22.00	38.00	36.00	**37.67**

(b) Goal-condition success rate (GC)

Type	LLM	Method	Base	Common	Complex	Long	Spatial	Visual	Overall
Sup.	–	HLSM	6.84	6.90	6.84	0.00	3.77	2.65	4.03
		FILM	8.87	6.30	4.96	9.55	0.87	5.83	6.45
		CAPEAM	12.10	8.66	8.26	17.59	5.22	5.83	10.42
ICL	LLaMA2	LLM-Planner	14.53	15.52	14.53	12.50	12.26	13.27	13.65
		FLARE	45.30	33.62	40.17	25.50	32.08	**46.02**	35.89
		AGR	**48.72**	**34.48**	**42.74**	**49.50**	**33.02**	29.20	**40.83**
	GPT-3.5	LLM-Planner	18.80	18.10	19.66	10.00	17.92	24.78	17.30
		FLARE	**51.28**	**43.97**	**43.59**	29.00	40.57	45.13	40.83
		AGR	50.43	**43.97**	33.33	**44.00**	**44.34**	**52.21**	**44.60**
	GPT-4o	LLM-Planner	27.35	24.14	30.77	24.00	35.85	30.09	28.09
		FLARE	53.85	48.28	**52.14**	38.00	**51.89**	46.90	47.33
		AGR	**57.26**	**56.03**	44.44	**42.50**	49.06	**48.67**	**48.89**

(c) High-level planning accuracy (HLP)

Type	LLM	Method	Base	Common	Complex	Long	Spatial	Visual	Overall
Sup.	–	HLSM	0.00	0.00	0.00	0.00	0.00	0.00	0.00
		FILM	0.00	0.00	0.00	2.00	0.00	0.00	0.33
		CAPEAM	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ICL	LLaMA2	LLM-Planner	4.00	2.00	10.00	0.00	2.00	4.00	3.67
		FLARE	40.00	16.00	32.00	2.00	30.00	22.00	23.67
		AGR	**46.00**	**26.00**	**36.00**	**8.00**	**40.00**	**26.00**	**30.33**
	GPT-3.5	LLM-Planner	20.00	8.00	20.00	2.00	16.00	18.00	14.00
		FLARE	**50.00**	22.00	34.00	0.00	32.00	42.00	30.00
		AGR	44.00	**42.00**	**40.00**	**4.00**	**48.00**	**46.00**	**37.33**
	GPT-4o	LLM-Planner	42.00	38.00	36.00	0.00	32.00	24.00	28.67
		FLARE	**56.00**	48.00	**56.00**	0.00	64.00	**46.00**	45.00
		AGR	**56.00**	**54.00**	52.00	**4.00**	**66.00**	**46.00**	**46.33**

C Task Planning Performance without Replanning

Given an instruction, AGR generates a task plan and does not revise the plan during task execution, whereas LLM-Planner and FLARE incorporate their own replanning mechanisms into the execution procedure. To assess task planning performance without the influence of replanning, we evaluate all ICL-based approaches without replanning.

As shown in Table C1, AGR achieves the best SR and GC on the ALFRED-based benchmarks and WAH-NL when all methods are evaluated without replanning. Compared with FLARE, AGR improves SR by 4.11% to 7.07% and GC by 2.47% to 7.16%. These results further demonstrate that AGR can guide LLMs to generate task plans that enable the agent to successfully complete tasks without modifying the plan during execution.

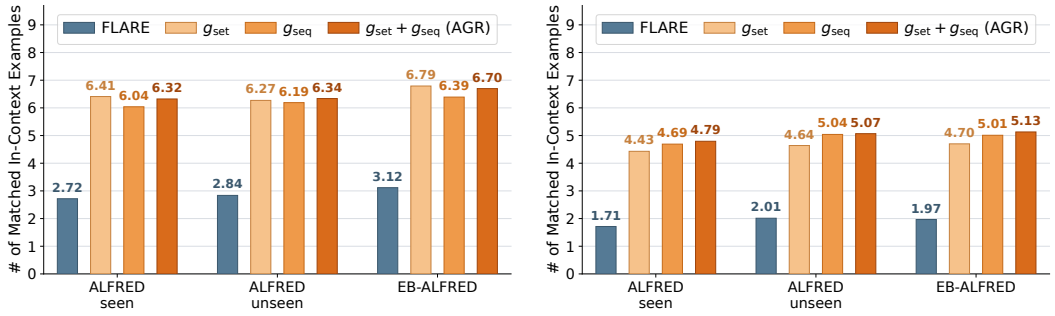
Table C1: Comparison of ICL-based approaches without replanning. For task plan generation, we use GPT-3.5 in all experiments. The best result in each column is highlighted in **bold**.

Method	ALFRED-seen		ALFRED-unseen		EB-ALFRED		WAH-NL	
	SR	GC	SR	GC	SR	GC	SR	GC
LLM-Planner	12.07	24.89	13.03	28.40	6.33	16.64	2.10	31.50
FLARE	27.44	37.93	30.09	41.46	28.67	40.05	4.10	38.20
AGR	**34.51**	**45.09**	**34.96**	**47.17**	**33.00**	**44.60**	**8.21**	**40.67**

D Analysis of Retrieved Examples

We analyze the effectiveness of AGR in retrieving examples that match the query in terms of the required actions on the ALFRED validation set and EB-ALFRED. Since ground-truth task plans are not provided for the ALFRED test set and WAH-NL, these benchmarks are excluded from this analysis. For each query, we count the number of examples among the top-9 retrieved examples that share the same action set or action sequence as the query and report the average over all queries.

As shown in Figure D1, AGR and its variants using only g_{set} or g_{seq} retrieve substantially more matching examples than FLARE on the ALFRED validation set in both seen and unseen environments, as well as on EB-ALFRED. When the projectors are used individually, g_{set} retrieves more examples sharing the same action set as the query, whereas g_{seq} retrieves more examples sharing the same action sequence. When both projectors are used together, AGR retrieves nearly as many examples sharing the same action set as g_{set} alone, while retrieving more examples sharing the same action sequence than g_{seq} alone, although the margin is modest. These results suggest that g_{set} and g_{seq} capture different aspects of the actions required by the query, and that using them together enables AGR to retrieve examples reflecting both aspects. The benefit of using both projectors is also reflected in the ablation results in Table 4, where AGR achieves the best or second-best SR and GC across all benchmarks among the compared variants.



(a) Action-set matching

(b) Action-sequence matching

Figure D1: Average number of retrieved in-context examples among the top-9 that match the query in terms of (a) action set and (b) action sequence.

E CODE AVAILABILITY

The code for the proposed method is attached as a zip file (`AGR_codes.zip`) for reproducibility.