Plant in Cupboard, Orange on Rably, Inat Aphone. Benchmarking Incremental Learning of Situation and Language Model using a Text-Simulated Situated Environment

Jonathan Jordan[1], Sherzod Hakimov[1], David Schlangen[1,2]

[1]Computational Linguistics, Department of Linguistics
University of Potsdam, Germany
[2]German Research Center for Artificial Intelligence (DFKI), Berlin, Germany
firstname.lastname@uni-potsdam.de

Abstract

Large Language Models (LLMs) serve not only as chatbots but as key components in agent systems, where their common-sense knowledge significantly impacts performance as language-based planners for situated or embodied action. We assess LLMs' incremental learning (based on feedback from the environment), and controlled in-context learning abilities using a text-based environment. We introduce challenging yet interesting set of experiments to test i) how agents can incrementally solve tasks related to every day objects in typical rooms in a house where each of them are discovered by interacting within the environment, ii) controlled in-context learning abilities and efficiency of agents by providing short info about locations of objects and rooms to check how faster the task can be solved, and finally iii) using synthetic pseudo-English words to gauge how well LLMs are at inferring meaning of unknown words from environmental feedback. Results show that larger commercial models have a substantial gap in performance compared to open-weight but almost all models struggle with the synthetic words experiments.

1 Introduction

Theoretical reasoning, which involves deriving factual conclusions from given premises, has been extensively studied in the context of large language models (LLMs) (Brown et al., 2020; Wei et al., 2022; Kojima et al., 2022). There has been less work focusing on incremental learning (Schlangen, 2023) of agents with regards to how easily they generalise to unknown environments. While theoretical reasoning can be evaluated using reference propositions, incremental learning presents unique challenges because the agent is required to observe consequences of actions in an environment (can be unfamiliar one) and learn how to interact step-by-step. There is significant interest in using LLMs for embodied AI tasks in robotics and simulations. (Ichter et al., 2022). Many of the existing benchmarks (WebArena (Zhou et al., 2024), ALFRED (Shridhar et al., 2021), AI2-THOR (Kolve et al., 2022), Balrog Arena (Paglieri et al., 2024)) include complex environments or require computational resources to execute vision-related tasks. In contrast, text-based interactive fiction (IF) environments such as TALES (Cui et al., 2025) or TextWorld (Côté et al., 2019), where a situation is described textually and updated in response to textual commands, make it possible to study reasoning abilities that go from goal and situation description to goal-directed action in greater isolation.

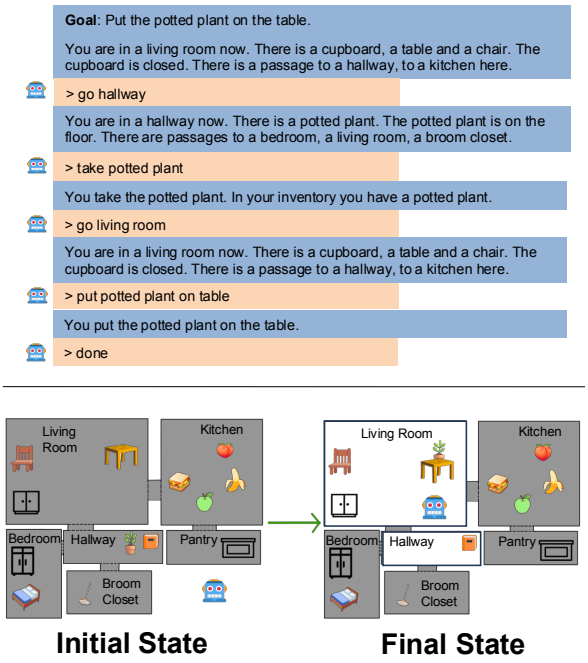

Figure 1: Sample representation of an episode in our game. The agent (an LLM) is given a task and randomly assigned to some room (living room). The environment provides feedback for every action (go, take, put, etc.) of an agent. The top part is how the game is played in a textual world. The bottom part is the visual representation of initial and final states, also indicating the knowledge gained by the agent.

In this paper, we focus on incremental learning capabilities of agents (LLMs) under controlled and uncontrolled conditions. Agents have to uncover the restrictions and rules of the task by interacting and taking the feedback (in-context learning) by environment into account. We also examine to what degree the underlying common-sense knowledge embedded in LLMs helps to navigate the tasks and how much of it is due to their generalisation capabilities. We expand this by introducing our text-based *AdventureGame*[1] environment to control the applicable level of common-sense world knowledge, for testing situated language environment understanding, spatial navigation, and instruction following abilities. We introduce various experiments to assess the effect of common-sense knowledge and generalisation capabilities where the agent (LLM acting on an environment) has to perform *home delivery* tasks with common objects within room types or has to infer the meaning of synthetic words by solely interacting and getting feedback from the environment. Figure 1 (top) shows an example of an interaction, while the bottom part is the visual representation of what has changed going from an initial state to the final one. Our contributions are as follows:

- A set of experiments based on an established object delivery task in a typical home setting, with varied levels of complexity, situated interaction task demands and information to be acquired.
- Synthetic words experiment to assess reliance on common-sense knowledge and incremental learning capabilities
- Quantitative and qualitative comparison of recent large language models (LLM) of varying sizes
- In-depth analysis to assess models' behaviour, examining good interactions and common failure cases.

2 Related Work

In-context Learning from Feedback: LLMs acquire situational information in IF environments through self-guided in-context learning (ICL) from observations and failure feedback. Testing LLM generalisability requires test instances unlikely to be seen during training to avoid reliance on memorized common-sense knowledge. *BlocksWorld* obfuscates block-manipulation words (Gragera and Pozanco, 2023; Valmeekam et al., 2024, 2023), while Eisenschlos et al. (2023) tested ICL by providing pseudo-English words with explanations to measure proper application in WinoGrad schema. Shinn et al. (2023) use "verbal reinforcement learning" with contextual feedback for environment interaction without weight training. Shi et al. (2024) found that smaller models rely on semantic priors while larger ones override mappings based on input context in label-flipping tasks. *AdventureGame* extends IF benchmarking with synthesized words in household tasks, from partial to fully synthetic non-function vocabulary.

Interactive Fiction (IF) Environments: These environments have been used to *train and compare agents* where they combine a specific set of agentic challenges: partial observability, large world state and action space, requiring exploration and common-sense reasoning, in addition to the text-only interaction and feedback. It has been shown that performance in text-only IF environments is transferable to embodied environments (Shridhar et al. (2021), Jansen (2021)). Interactive fiction (IF) environments have also been used to *compare LLM performance* in regards to their world modelling, task solving and planning capabilities (Wang et al. (2022), Tan et al. (2023), Tsai et al. (2023), Ma et al. (2024), Gioacchini et al. (2024)). Jericho (Hausknecht et al., 2020) provides a framework for classic IF games, which remain challenging for LLMs due to complexity and humorously nonsensical solutions. *AdventureGame* condenses core IF challenges while avoiding overly difficult genre aspects. TextWorld (Côté et al., 2019) extends Jericho for large-scale simplified IF generation and agent benchmarking. *AdventureGame* enables similar generation but records detailed interaction scores and exploration data. ALFWorld (Shridhar et al., 2021) combines 3D-embodied simulation with text-only pre-training, demonstrating text interaction transferability to multimodal domains. TALES (Cui et al., 2025) benchmarks LLM reasoning by combining the above IF environments with *Simon Says* and ScienceWorld (Wang et al., 2022).

AdventureGame has a unique focus on text-only household object placement tasks by requiring the tested LLMs to perform in-context learning from minimal natural language feedback to assess the core reasoning, instruction following, situated ac-

[1]Source code: https://github.com/clembench/clembench/tree/main/adventuregame (inspired by the "Colossal Cave Adventure" game by W. Crowther: https://ifdb.org/viewgame?id=fft6pu91j85y4acv)

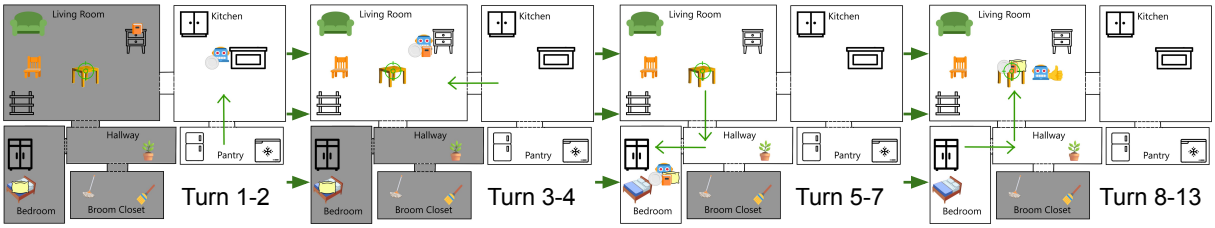

Figure 2: Simplified illustration of *AdventureGame* interaction. The agent is controlled by an LLM. The task is *put the plate, book and pillow on the table* (marked by green crosshair). The agent starts in the pantry. Unexplored rooms are gray. The agent first goes to the kitchen, *takes the plate from the counter* (Turn 1-2), then *goes to the living room, takes the book* (Turn 3-4). Next, it *goes to the hallway*, and from there *to the bedroom where it picks up the pillow* (Turn 5-7), then it *returns to the living room, puts the carried objects on the table* (Turn 8-13).

tion selection, and, most importantly, the effects of synthetic vocabulary to test generalization beyond training-embedded common-sense knowledge.

3 *AdventureGame*: IF Environment

In this section, we describe the details of the game. A simplified illustration is given in Figure 2, where an LLM controls an agent to perform the given task. We implemented the game using the clembench framework (Chalamalasetti et al., 2023) (prompts are provided in Appendix D).

3.1 Task Definition

Interactive Fiction games can be considered partially observable Markov Decision processes (Kaelbling et al., 1998), with the player having only partial information about the game's world state from textual observations. The player must first discover all task-relevant locations and objects, making exploration as much a part of solution strategies as effectively executing a plan for manipulating objects to reach the game's goals. To do so successfully, the player has to model environment states (like room connections and locations of objects), as well as the large action space and state transitions resulting from actions, especially over multiple turns.

The task is defined by the tuple $\langle G, S, A, O, T \rangle$ with a set of goal states G, state space S, valid action space A, observation space O (including IF interpreter feedback), and transition function $T : S \times A \rightarrow S$.

An agent with policy π makes predictions at time step t based on goals in G and memory $m_t = \{o_j, a_j, o_{j+1}, a_{j+1}, ...o_t\}, 0 \leq j < t$, which is a sequence of actions and observations. This agent trajectory $\tau = [s_0, a_0, s_1, a_1, ...s_t]$ is formulated by policy and environmental state transitions as below where a time step t is one turn of the game:

$$p_\pi(\tau) = p(s_0) \prod_{t=0}^{T} \pi(a_t|G, s_t, m_t)\tau(s_{t+1}|s_t, a_t)$$

All state transitions are deterministic, so only the actions generated by the LLM as text commands determine the agent's trajectory. Observations are likewise deterministic. We assume that the language model models an underlying policy, tapping into the capabilities under examination.

3.2 Interaction

World state, representing a temporary subset of S, is stored as a set of fact tuples, describing both mutable states of entities and immutable states. Mutable states are `at`, `in`, `on`, `open` and `closed`. Immutable states describe entity types and categories and are used as conditions for actions (see Table 5).

Actions are defined in the PDDL (Fox and Long, 2003) format, covering the state changes they enact (representing T). The action definition also contains a Lark parser grammar snippet that is used to form the parse-able input command grammar, feedback templates for success and individual failures (covering O) and a Clingo (Gebser et al., 2017) encoding snippet of the state changes of the action for optimal solution solving (representing T as well).

The player is not given a list of currently available actions but rather has to model the action space itself. Movement is only allowed to connected rooms (unlike in Basavatia et al. (2024) and others, which allow "teleportation").

Turn Limit: The interaction is limited to 50 turns, and reaching the limit is recorded as aborting that episode.

Formatting: An output produced by LLM has to follow the format below (starting with the prompt symbol >), e.g. > *go hallway*:

```
> "Natural language command"
```

The episodes where generated outputs that do not follow this formatting are aborted.

Parsing & State Change: The interpreter attempts to parse the command, and if it passes, the corresponding action is performed in the resolution phase. State change conditions are checked, and any resolution failure stops the process. If state change conditions are fulfilled, the game world state is updated accordingly, removing and adding facts in the world state. All changes in the world state are recorded. Lastly, the interpreter checks if changed states are part of the goal state set G and returns achieved goal states G_a, textual feedback $o \in O$ and any failure that might have occurred in processing the input command to be recorded.

The player ends an episode by generating the "done" command. Once the episode ends, goal states achieved as of the last turn are recorded—meaning that goal states achieved intermittently are not considered for metrics.

Exploration Tracking: All commands, state changes, observed locations and objects are recorded for each turn, including errors while executing the commands. We label each action (inspired by Kirsh and Maglio (1994)) for being *epistemic* and *pragmatic*. An action is *epistemic* when it improves a player's perception of the game situation without directly progressing towards the goal[2] and *pragmatic* when the action directly works towards reaching the goal. Then, we calculate *epistemic gain* by counting how many world state facts the player newly observes as a result.

4 Experimental Setup

4.1 Game Instances

We create different set of instances to test incremental learning under uncontrolled (Basic, Synthetic words) and controlled (Pre-exploration, Inventory limit) conditions.

4.1.1 Basic

The core task in our experiments is a *delivery task* where the agent is expected to deliver three objects to target receptacles in a typical home environment.
Objects and rooms: The basic instances contain 22 everyday household objects (e.g. plant, book), six common room types (e.g. kitchen, hallway) and seven staple IF actions (e.g. go, take). The complete lists are in Appendix C.
Difficulty levels: easy & hard. The *easy level* means that goal objects are not inside closed containers (easily accessible, immediately observable once the agent enters the room) and are located near the target receptacle in the initial world state. The *hard level* means that goal objects are initially hidden in closed containers (e.g. cupboards, closets), where each needs to be delivered to a different target. There are also longer paths between initial goal object locations and their targets. Our hypothesis is that it is challenging for models to navigate multi-step actions to reach a goal because hidden objects require epistemic actions, and far away objects require keeping the objective in mind over several steps (increases the number of steps to reach goals).

4.1.2 Inventory Limit

Using the same settings described above, we create another configuration by setting a limit on how many objects can be carried by the agent at a time. We set the limit as *two* objects. We want to analyse whether models lean towards strategic use of the resource "inventory" and are efficient when they have the limit. The set of objects and rooms in instances used here are the same as in *Basic* experiment.

4.1.3 Pre-Exploration

We aim to test the effect of additional context at the start of the interaction. We provide the information about the rooms and which objects are there located there to the agent. This effectively inserts the layout of rooms needed for correct navigation and locations of task objects (or the containers holding them in 'hard' instances) into context. As this automated interaction is formatted like any correct interaction input and feedback, the pre-exploration sequence can be considered as a *few-shot in-context learning*. Pre-exploration sequences are six to eight "go" actions. We want to analyse whether models get more efficient (compared to the *Basic* experiment) in completing the goals. The set of objects and rooms in instances used here are the same as in *Basic* experiment.

4.1.4 Synthetic Words

We create another experiment set by replacing words in the home delivery task with pseudo-English words using scripts by Eisenschlos et al.

[2] Kirsh and Maglio's example is moving a Tetris tile to the leftmost position to be sure about its position instead of moving it towards the location that is most beneficial to put it down in.

(2023) while retaining an unaltered basic set of common IF actions. The goal here is to assess whether agents can solve the task incrementally by relying solely on the feedback from the environment. These words are unknown to any LLM. Thus, embedded common-sense knowledge does little affect performance and focuses more on generalisation abilities. We defined three difficulty levels: easy, medium & hard.

Easy level: Three objects and one action word are replaced with pseudo-English words. A short explanation of the replaced action word is provided. For example, we apply the following replacements: *bedroom* → *enticed*, *book* → *decte*, *shelf* → *stord*. The verb *put* is replaced with *aphon*, and the explanation "In addition to common actions, you can aphon. To aphon is to physically place something somewhere." is provided in the initial prompt. The correct command to *put the book on the shelf* is then > *aphon decte (on) stord*.

Medium level: Nine objects and three action words are replaced with pseudo-English words. The replaced synthetic actions words are listed in the initial prompt without explanations, such as "In addition to common actions, you can inate, pante and eness". In this case, *eness* replaces *close*, *pante* replaces *put* and *inate* replaces *open*.

Hard level: This variant uses pseudo-English words for all entity and room nouns, state adjectives and action verbs, with the task being to change the states of entities using the verbs, removing the possibility of relying on common-sense knowledge. As we expected, the ubiquity of synthetic words would provide great difficulty in itself, but these instances contain only four rooms in a simple linear arrangement. Object state interaction complexity ranges from single state change (> *mator subst* directly resulting in *the subst is now dent*, fulfilling a goal state), via binary state sets (> *unbal diale* changes the *unsust-able*' *diale* object from being *unsust* to being *exper*), to state sets with three states, requiring the use of two actions in the correct order to bring an object into a target state.

Our intention here is to validate models' reliance on common-sense knowledge seen during their training phases. These experiments require the models to apply in-context learning to uncover the meanings of synthetic words by interacting with the environment and getting feedback on their actions.

4.1.5 Number of Instances

In total, we have nine experiments for each variant described above: Basic-easy, Basic-hard, Basic-easy-limit-two, Basic-hard-limit-two, Basic-easy-preexplore, Basic-hard-preexplore, Synthetic-words-easy, Synthetic-words-medium and Synthetic-words-hard. Each experiment has *16* instances, corresponding to *144* instances in total.

4.2 Metrics

Framework-specific metrics: The clembench framework includes two main metrics: *Played* & *Quality Score*. The game finishes successfully only when a model produces > *done* as the last action and all goals have been achieved. The game is *aborted* when a model does not follow formatting requirements (see Section 3.2) or reaches the *maximum turn limit*, which is 50. Played is the ratio of instances that were not aborted. Quality Score measures how many episodes have all their goal states reached at the end. Producing the > *done* action command without achieving all goal states is considered a *lost* episode. In cases where all goal states are achieved and > *done* is also generated, then the episode is *successful*.

Finally, to rank the benchmarked models, the framework includes a metric called *clemscore*, the macro-average *Quality Score* multiplied by the macro-average proportion of *Played* games across all experiments.

Game-specific metrics: We have a specific metric to keep track of achieved goals. It is the ratio between achieved goal states G_a and all goal states G: Goal Success Rate (GSR) $= \frac{|G_a|}{|G|} \times 100$.

4.3 Evaluated Models

We evaluated open-weight and commercial models (with *temp=0*). We included recent commercial models such as: *o3-mini* (Jan '25), *GPT-4o* (Aug '24) *Claude-3-7* (Sonnet, Feb '25). We also included recent open-weight models: *Llama-3.1* (8B, 70B) (Grattafiori et al., 2024), *Llama-3.3* (70B), *Qwen2.5* (Coder-32B, 72B) (Qwen et al., 2025), and *Deepseek-v3* (DeepSeek-AI et al., 2024). We used the APIs of the respective commercial models. We ran open-weight models on two NVIDIA A100 GPUs. Deepseek-v3 was run via the OpenRouter API.

Model	clem score	Quality Score	% Played	Goal Rate
Claude-3.7	**86.4**	**88.2**	**97.9**	**90.5**
o3-mini	68.7	79.2	86.8	85.9
GPT-4o	52.7	56.2	93.8	75.9
Llama-3.1-70B	43.8	49.2	89.2	66.8
Llama-3.3-70B	40.1	43.3	92.5	64.4
Deepseek-v3	39.8	46.5	85.4	64.8
Qwen2-72B	15.9	28.8	55.4	47.5
Qwen2.5-32B	12.0	25.0	47.9	47.8
Qwen2.5-72B	11.2	21.2	52.9	44.9
Llama-3.1-8B	8.2	20.6	39.9	35.7

Table 1: Overall benchmark scores for models.

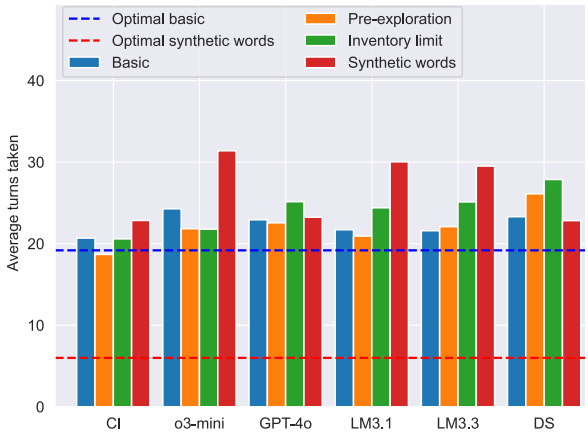

Figure 3: Average number of turns taken by each model for each experiment category and average optimal number of turns for each category are also included (dotted lines).

5 Results

5.1 Overall Comparison

Table 1 shows the main scores for the benchmarked models. Larger models achieve higher quality scores and better conform to the prompted output format (yielding higher % Played). Most commercial models achieve higher scores than open-weight models (8.9 points between *GPT-4o* and *Llama-3.1-70B*), with *Claude-3.7* scoring higher than the next best model by at least 9 points on the three major metrics. Another observation is that all high-ranking models can play the game (follow the instructions) but lack performance in solving the task. Next, we analyse the cases deeper to uncover which factors contribute to low scores.

5.2 In-depth Analysis

Table 2 presents the *Quality* and *% Played* values for each experiment. We selected the top-performing six models. The results for other models can be found in Table 3. Next, we break down the results across each experiment.

Basic Navigation & Task Solving: In the *easy* setting, most models seem to get somewhat adequate performance (>=50). Notably, among the compared top six models, *o3-mini* is the only model without a full % Played score in this experiment, leading to it not matching the performance of the best model, *Claude-3.7*. However, *o3-mini* achieves the highest scores in the *hard* setting, with it and especially *Claude-3.7* being impacted far less by the increased task complexity.

Effects of Pre-Exploration: This experiment aims to see whether models increase their performance since initial rooms and certain objects and containers are revealed. When comparing the results in the *Basic* experiment. The unexpected behaviour is with *o3-mini* and *GPT-4o* on the *hard*, where their performance worsens despite the provided pre-exploration context. Figure 3 visualises the average number of turns it takes for each experiment category. The figure also provides the optimal number of turns (averaged). We can clearly observe that pre-exploration indeed helps the high-scoring models (Claude-3.7) to be more efficient (requiring less number of turns) and yield better performance when compared with the *Basic* experiments.

Inventory Limit: When compared with the *Basic* experiments, the performance of *Claude-3.7* and *o3-mini* models get better while others yield similar performance. Figure 3 visualises similar patterns where they do not need more turns than the *Basic* experiments. *o3-mini* even reduces the number of turns with the limit. It suggests that these models can navigate the inventory limit while retaining or even increasing performance. However, lower-performing models (GPT-4o, LLama models, Deepseek-v3) tend to take much longer than the *Basic* experiment. It shows that these models get affected by the inventory limit not much on the general task solving but having lower efficiency.

Synthetic Words: In the *easy* setting, the smaller models (*Llama-3.1-8B*, *Qwen2.5-32B-Coder*) yield the worst performance showing that the required incremental learning abilities are severely lacking in smaller models. All commercial models *o3-mini*, *GPT-4o*, *Claude-3.7*, *GPT-4o* show an improvement over the *Basic* experiment results. *Claude-3.7* even achieves full scores, making it the best model in this experiment.

The larger amount of synthetic words in the *medium* setting does lead to a drastic drop (expected by the design of the experiment) in scores

Experiment		Claude-3.7		o3-mini		GPT-4o		LM-3.1		LM-3.3		DS-v3	
		Q	P	Q	P	Q	P	Q	P	Q	P	Q	P
Basic	easy	75.0	100	75.0	87.5	68.7	100	62.5	100	56.2	100	62.5	100
	hard	81.2	93.7	87.5	93.7	43.7	87.5	31.2	87.5	18.7	93.7	56.2	87.5
Pre-Exploration	easy	75.0	100	75.0	93.7	87.5	100	68.7	100	68.7	100	93.7	100
	hard	93.7	93.7	75.0	75.0	37.5	87.5	50.0	93.7	31.2	100	37.5	56.2
Inventory Limit	easy	81.2	100	87.5	100	68.7	100	62.5	100	50.0	100	68.7	100
	hard	93.7	93.7	93.7	93.7	31.2	87.5	37.5	87.5	31.2	93.7	31.2	56.2
Synthetic Words	easy	100	100	87.5	100	81.2	100	68.7	93.7	43.7	93.7	43.7	100
	medium	100	100	43.7	43.7	43.7	81.2	37.5	50.0	18.7	50.0	25.0	81.2
	hard	93.7	100	87.5	93.7	43.7	100	12.5	87.5	43.75	93.75	0.0	87.5

Table 2: Detailed results across different experiments. Only high performing six LLMs were selected. The values are *Quality Score* (Q) and *% Played* (P) separated by the / (slash sign) for each experiment. *LM-3.1* → Llama-3.1-70B, *LM-3.3* → Llama-3.3-70B, DS-v3 → Deepseek-v3

compared to the *easy* experiment, lowering scores by at least half. However, *Claude-3.7* still yields the perfect scores, again.

Performance impact diverges between models for the *hard* setting, where we expected the task to be more challenging for the models than the *medium* setting. While *Llama-3.1-70B* and *DeepSeek-v3* (0.0 Quality Score) do worse when confronted with many synthetic words, *o3-mini* performs almost as well as in the *easy* experiment and *Claude-3.7* achieves high performance again (fails only in a single episode).

Figure 3 shows that none of the models are that efficient (requiring more turns than the optimal solution) as they were with other experiments. It suggests that, as expected, the models need more turns to infer the meaning of words. Claude-3.7 is not only the best performing model but also the most efficient one in these experiments.

5.3 Qualitative Analysis

Navigation and self-correction: All models make navigation errors despite the observation feedback mentioning passages to all connected rooms every time the player enters a room. While lower-performing models repeat this type of failure more in individual episodes, higher-performing models acquire the game's rule that allows movement only to connected rooms and self-correct their navigation after receiving feedback. This type of failure most often occurs in the turn after a task object has been picked up or delivered. Figure 4 illustrates how *Claude-3.7* attempts to "> go to hallway" from the unconnected bedroom (red arrow) and is told that this is not possible. It then respects the connection requirement for the rest of the episode and goes to the *broom closet* to *take the mop* and further *to the pantry with the freezer* without attempting to go to these known rooms directly.

Insufficient exploration: Figure 5 illustrates how *Qwen2.5-Coder-32B* regularly checks connected rooms, e.g. *hallway*, which largely contributes to its performance. However, it does not do this thoroughly enough, missing the *table in the pantry* and incorrectly placing the task objects on the side *table*, as most models do.

Synthetic words & in-context learning: Figure 6 shows a typical behaviour of well-performing models in the synthetic words experiment: After attempts to take the book (regardless of it not being needed for the task), *o3-mini* tries the synthetic word actions provided in the initial prompt. The environment feedback contains the information necessary to learn that "aling" means to "take", "vater" means to "open" and "nogia" means to "close". Later in the same episode, shown in Figure 7, *o3-mini* fails to use the synthetic verbs properly. It eventually produces the right action "vater imped", allowing it to fulfil one of the three task goals, but the episode ends due to reaching the turn limit.

6 Discussion

Small models lack generalisation capabilities: models that perform worst in the synthetic word experiments lack the ability to generalise beyond the content of their training data, with the introduction of even a few unseen token sequences in context disrupting their performance.

Large models show individual differences for common-sense knowledge vs. in-context learning (ICL): the mixed performance between larger models indicates substantial differences in ICL and generalisation capabilities between individual models. While these models do not fail like the smaller ones, they are distracted by encountering synthetic words, eventually fail to solve the tasks.

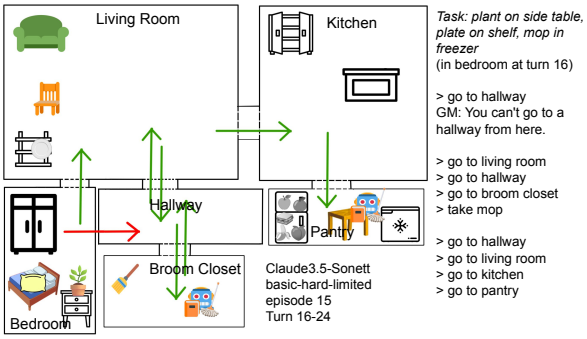

Figure 4: *Claude-3.7* correcting navigation.

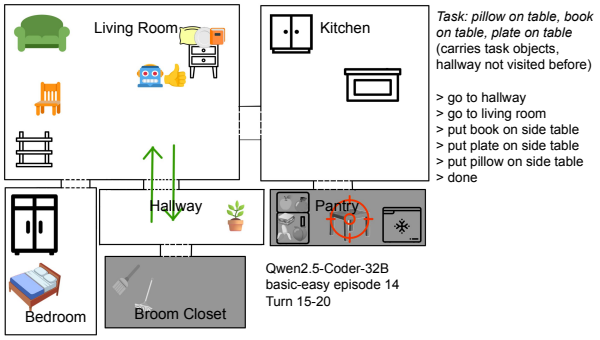

Figure 5: *Qwen2.5-32*B exploring insufficiently.

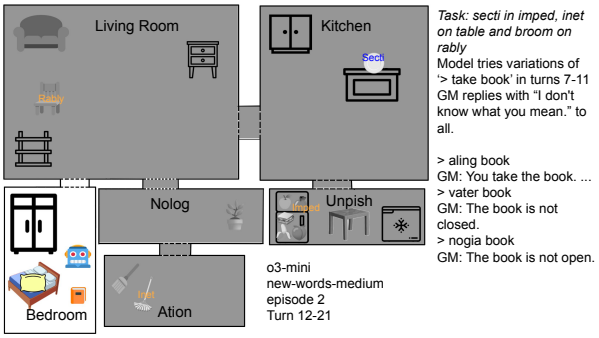

Figure 6: *o3-mini* exploring synthetic actions.

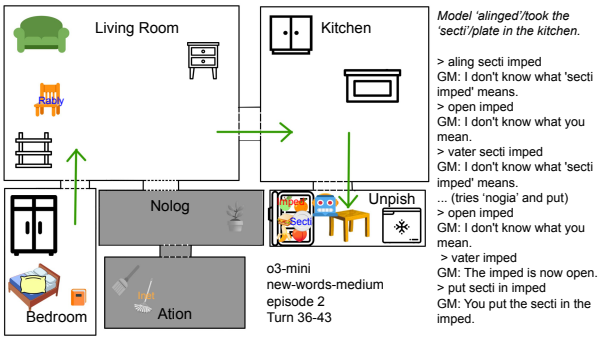

Figure 7: *o3-mini* misapplying synthetic actions.

Automatic provision of information can be detrimental: the negative impact of the navigation head-start in the pre-exploration experiments compared with the basic experiments is unexpected, as it should provide a large part of the necessary exploration and the situation state information needed to solve the task better. However, we assume that the observed performance is an effect of ICL as well: The pre-exploration sequence essentially serves as few-shot examples *to not interact with task-relevant objects* when they are encountered, which is opposed to the behaviour the tested models show without it, interacting with task objects immediately and thus directly pursuing task goals.

Situation complexity is more important than words: the difference in scores in the *easy* and *medium* variants suggests that strong common sense knowledge impedes the ability to learn new information for all models except *Claude-3.7*, which excels in this acquisition. The higher scores some models achieve in the *hard* synthetic words experiment are due to the relatively low complexity of the situation, requiring less navigational exploration and lacking task objects that need to be revealed through interaction. Models that can learn the meaning of arbitrary synthetic words from the feedback provided when interacting with these words can thus solve these tasks better, as shown by the high scores of *o3-mini* and *Claude-3.7*.

Human baseline: we conducted a small study and played five instances from each experiment and measured 100 points in all metrics while Claude-3.7 scoring very close to it (see Table 4).

7 Conclusion

Our experimental results indicate that performance increases with model size, progressing from generally bad situation modelling in smaller models to a middle ground of good situation modelling but frequent interaction failures, to only a few larger models fulfilling the given task in more than two thirds of cases. Incremental learning abilities of models were tested even further with the synthetic words experiments and showed that the gap between commercial and open-weight models is still quite big. Models relying more on their embedded common-sense knowledge performed worse and were less efficient that those models that are capable of incrementally figuring out the task and applying in-context learning.

Limitations

The current study is restricted to only English in its current state. While we have yet to do this, translating the prompts and adapting the underlying

grammar entries is possible for other languages, too.

The performance we measured here may not transfer to other modalities with more sophisticated demands, like visually or physically embodied agents or robots. Shridhar et al. (2021) found that while training in text-only environments is faster and less resource-intensive than training in the AI2Thor framework, agents trained in text-only environments struggled to adapt to the requirements of more complex embodiment properly.

Ethics Statement

In academic research, using paid proprietary APIs with underlying models about which little is known (training data, model architecture) is less than ideal. Currently, the models benchmarked here are the high-performing ones that are commercially used. We hope that more open models with high performance will be released soon and that proper research can be done on them.

Models that may be used in agents, possibly embodied as robots, being able to acquire replacement lexicons or entirely new semantics simply through ICL bears the possibility to circumvent safety measures from model creators and providers. We do not condone any such attempts based on our findings on the amenability of certain models to this kind of manipulation, commonly known as jailbreaks, specially when LLM agents are capable of interacting with computer systems or controlling robots.

References

Shreyas Basavatia, Keerthiram Murugesan, and Shivam Ratnakar. 2024. Starling: Self-supervised training of text-based reinforcement learning agent with large language models. *Preprint*, arXiv:2406.05872.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, and et al. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Kranti Chalamalasetti, Jana Götze, Sherzod Hakimov, Brielen Madureira, Philipp Sadler, and David Schlangen. 2023. Clembench: Using game play to evaluate chat-optimized language models as conversational agents. *Preprint*, arXiv:2305.13455.

Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler. 2019. Textworld: A learning environment for text-based games. In *Computer Games*, pages 41–75, Cham. Springer International Publishing.

Christopher Cui, Xingdi Yuan, Ziang Xiao, Prithviraj Ammanabrolu, and Marc-Alexandre Côté. 2025. Tales: Text-adventure learning environment suite. *arXiv preprint arXiv:2504.14128*.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, and et al. 2024. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

Julian Martin Eisenschlos, Jeremy R Cole, Fangyu Liu, and William Cohen. 2023. Winodict: Probing language models for in-context word acquisition. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 94–102.

Maria Fox and Derek Long. 2003. Pddl2. 1: An extension to pddl for expressing temporal planning domains. *Journal of artificial intelligence research*, 20:61–124.

Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub. 2017. Multi-shot ASP solving with clingo. *CoRR*, abs/1705.09811.

Luca Gioacchini, Giuseppe Siracusano, Davide Sanvito, Kiril Gashteovski, David Friede, Roberto Bifulco, and Carolin Lawrence. 2024. Agentquest: A modular benchmark framework to measure progress and improve llm agents. *arXiv preprint arXiv:2404.06411*.

Alba Gragera and Alberto Pozanco. 2023. Exploring the limitations of using large language models to fix planning tasks. In *ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, and et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Matthew Hausknecht, Prithviraj Ammanabrolu, Marc-Alexandre Côté, and Xingdi Yuan. 2020. Interactive fiction games: A colossal adventure. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7903–7910.

Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, and et al. 2022. Do as I can, not as I say: Grounding language in robotic affordances. In *Conference on Robot Learning, CoRL 2022, 14-18 December 2022, Auckland, New Zealand*, volume 205 of *Proceedings of Machine Learning Research*, pages 287–318. PMLR.

Peter A Jansen. 2021. A systematic survey of text worlds as embodied natural language environments. *arXiv preprint arXiv:2107.04132*.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134.

David Kirsh and Paul Maglio. 1994. On distinguishing epistemic from pragmatic action. *Cognitive Science*, 18(4):513–549.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, Aniruddha Kembhavi, Abhinav Gupta, and Ali Farhadi. 2022. Ai2-thor: An interactive 3d environment for visual ai. *Preprint*, arXiv:1712.05474.

Lark parser. 2024. [link].

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. 2024. Agentboard: An analytical evaluation board of multi-turn llm agents. *arXiv preprint arXiv:2401.13178*.

Davide Paglieri, Bartłomiej Cupiał, Sam Coward, Ulyana Piterbarg, Maciej Wołczyk, Akbir Khan, Eduardo Pignatelli, Łukasz Kuciński, Lerrel Pinto, Rob Fergus, Jakob Nicolaus Foerster, Jack Parker-Holder, and Tim Rocktäschel. 2024. Balrog: Benchmarking agentic llm and vlm reasoning on games. *arXiv preprint arXiv:2411.13543*.

Qwen, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, and et al. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

David Schlangen. 2023. What A situated language-using agent must be able to do: A top-down analysis. *CoRR*, abs/2302.08590.

Zhenmei Shi, Junyi Wei, Zhuoyan Xu, and Yingyu Liang. 2024. Why larger language models do in-context learning differently? In *Forty-first International Conference on Machine Learning*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pages 8634–8652. Curran Associates, Inc.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. 2021. Alfworld: Aligning text and embodied environments for interactive learning. *Preprint*, arXiv:2010.03768.

Qinyue Tan, Ashkan Kazemi, and Rada Mihalcea. 2023. Text-based games as a challenging benchmark for large language models.

Chen Feng Tsai, Xiaochen Zhou, Sierra S Liu, Jing Li, Mo Yu, and Hongyuan Mei. 2023. Can large language models play text games well? current state-of-the-art and open questions. *arXiv preprint arXiv:2304.02868*.

Karthik Valmeekam, Matthew Marquez, Alberto Olmo, Sarath Sreedharan, and Subbarao Kambhampati. 2023. Planbench: An extensible benchmark for evaluating large language models on planning and reasoning about change. In *Advances in Neural Information Processing Systems*, volume 36, pages 38975–38987. Curran Associates, Inc.

Karthik Valmeekam, Kaya Stechly, and Subbarao Kambhampati. 2024. LLMs still can't plan; can LRMs? a preliminary evaluation of openAI's o1 on planbench. In *NeurIPS 2024 Workshop on Open-World Agents*.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. 2022. Scienceworld: Is your agent smarter than a 5th grader? *arXiv preprint arXiv:2203.07540*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. 2024. Webarena: A realistic web environment for building autonomous agents. *Preprint*, arXiv:2307.13854.

A Additional Results

Figure 8 shows percentages of failures by processing phase.

Figure 9 shows percentages of entity-related failures. o3-mini and GPT-4o do not have inventory limit failures, while Qwen2.5-32B and Qwen2.5-72B have low amounts.

B Human comparison results

An annotator who has background in computational linguistics played the first five instances of the experiments, in classic IF fashion using the clemcore terminal. Table 4 shows overall scores for the tested models and human. Figure 11 shows the average number of turns taken to solve the first five episodes by experiment category, including human data.

Experiment		Qwen2-72B	Qwen2.5-32B	Qwen2.5-72B	LM-3.1-8B
Basic	easy	31.2/68.7	56.2/93.7	18.7/50.0	37.5/62.5
	hard	6.2/31.2	12.5/50.0	6.2/25.0	0.0/31.2
Pre-Exploration	easy	68.7/93.7	56.2/81.2	43.7/75.0	50.0/100
	hard	31.2/56.2	0.00/25.0	6.2/43.7	6.2/43.7
Inventory Limit	easy	56.2/87.5	25.0/75.0	43.7/93.7	31.2/75.0
	hard	25.0/56.2	12.5/37.5	6.2/31.2	12.5/18.7
Synthetic Words	easy	31.2/68.7	12.5/31.2	18.7/50.0	0.0/6.2
	medium	0.0/18.7	0.0/6.2	0.0/18.7	0.0/0.0
	hard	6.2/75.0	6.2/37.5	0.0/81.2	6.2/6.2

Table 3: Detailed results across different experiments for LLMs not shown in Table 2. The values are *Quality Score*/% *Played* for each experiment. *LM-3.1-8B* → Llama-3.1-8B

Model	clem score	Quality Score	% Played	Goal Rate
Human baseline	100.0	100.0	100.0	100.0
Claude-3.7	97.8	100.0	97.8	99.3
o3-mini	73.2	86.7	84.4	91.9
GPT-4o	51.4	88.9	57.8	78.5
Llama-3.1-70B	51.0	95.6	53.3	75.6
Llama-3.3-70B	45.6	93.3	48.9	71.1
DeepSeek-V3	40.2	82.2	48.9	66.7
Qwen2-72B	26.3	62.2	42.2	54.8
Qwen2.5-72B	17.3	55.6	31.1	55.6
Qwen2.5-32B	14.2	53.3	26.7	50.4
Llama-3.1-8B	11.3	42.2	26.7	43.7

Table 4: Overview results from the first five episodes of each experiment (total 45) including human data. The annotator has a background in computational linguistics and participated in the study voluntarily.

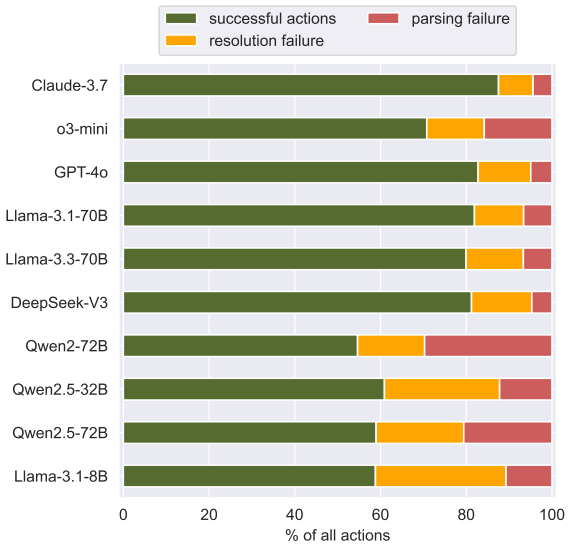

Figure 8: Successful and failed action quotas by IF interpreter processing phase.

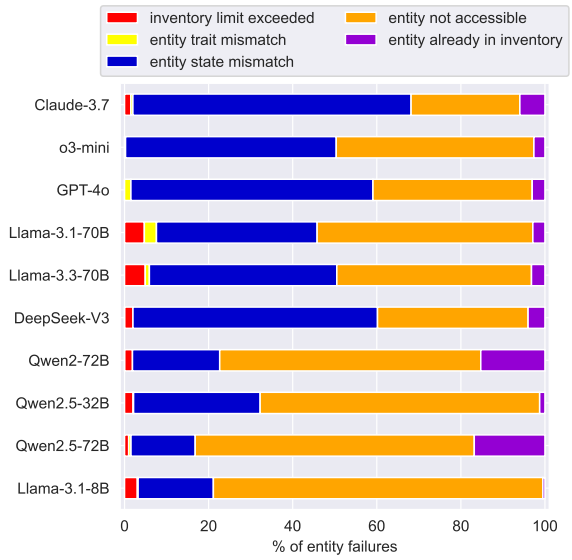

Figure 9: Percentages of selected entity-related failures for all tested models.

C IF Actions, Objects & Rooms

All experiment instances have the core entities "player" and "inventory" needed for the core AdventureGame IF interpreter. These and the 'floor' entity can not be replaced for the synthetic words *easy* and *medium* experiments.

The basic home domain has the following entities (with possible location rooms in brackets): Table (kitchen, living room), side table (living room, bedroom), counter (kitchen), refrigerator (kitchen, pantry), cupboard (kitchen), wardrobe (bedroom), shelf (kitchen, pantry, living room), freezer (pantry), potted plant (living room, hallway, bedroom), chair (living room), bed (bedroom),

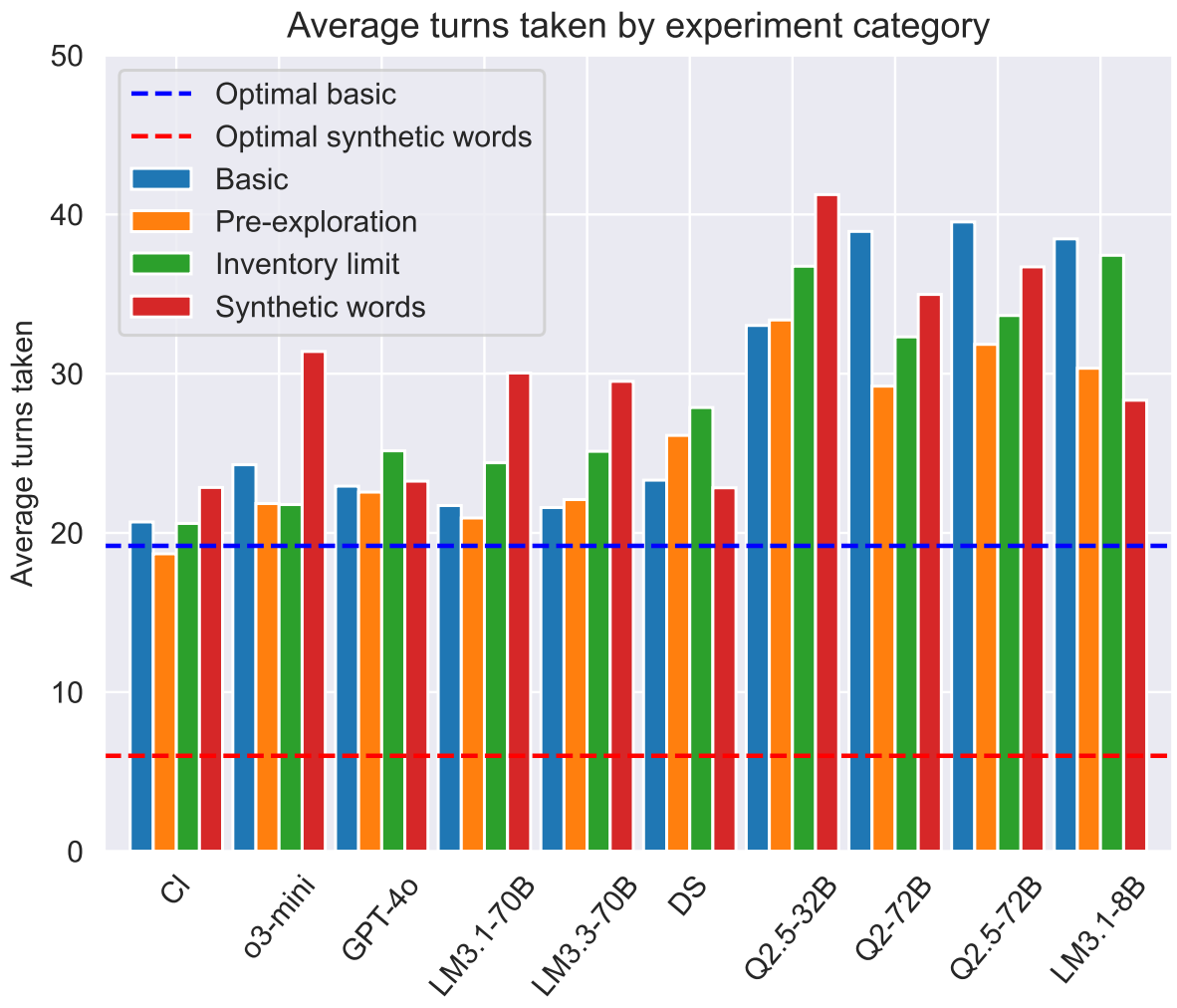

Figure 10: Average number of turns taken by each model for each experiment category. The averaged optimal number of turns for each experiment category is also given as dotted lines. Dotted blue lines for all basic, inventory limit and pre-exploration, red lines is for synthetic words experiments.

couch (living room), broom (broom closet), mop (broom closet), sandwich (kitchen, pantry), apple (kitchen, pantry), banana (kitchen, pantry), orange (kitchen, pantry), peach (kitchen, pantry), plate (kitchen), book (living room, bedroom), pillow (bedroom).

The following entities are "supports", allowing "movable" objects to be placed "on" them: Table, side table, counter, shelf, chair, bed, couch. The following entities are 'containers', allowing "movable" objects to be placed "in" them and objects in them being accessible if they are 'open': Refrigerator, cupboard, wardrobe, freezer.

The following entities are "movable": Potted plant, broom, mop, sandwich, apple, banana, peach, plate, book, pillow.

The basic home domain has the following rooms (with possible adjacent rooms in brackets): Kitchen (pantry, living room, hallway), pantry (kitchen, hallway), hallway (kitchen, pantry, living room, broom closet), living room (kitchen, hallway), broom closet (hallway) and bedroom (living room, hallway). All can be replaced for the synthetic words *easy* and *medium* experiments.

Table 5 lists the actions defined for all basic experiments, including which can be replaced for the synthetic words *easy* and *medium* experiments.

D Initial Prompts

Prompt template for 'basic' and pre-exploration variant instances:

```
You are playing a text adventure game. I
will describe what you can perceive in
the game. You write the single action you
want to take in the game starting with >.
Only reply with an action.
```

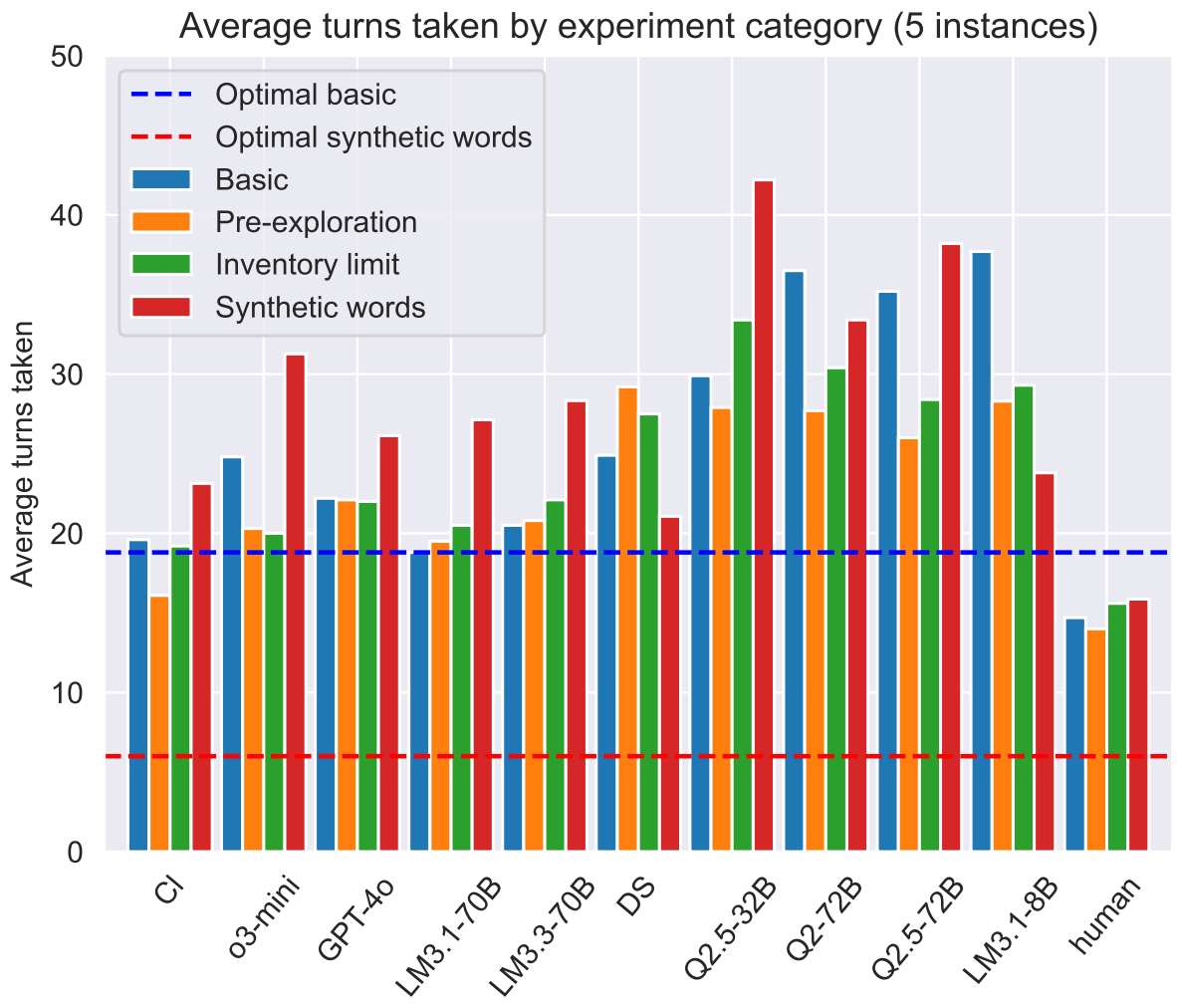

Figure 11: Average number of turns taken for the first five instances by each model and human for each experiment category. The averaged optimal number of turns for each experiment category is also given as dotted lines. Dotted blue line for basic, inventory limit and pre-exploration, red line is for synthetic words experiments.

```
For example:
> open cupboard

Your goal for this game is: $GOAL$

Once you have achieved your goal, write
"> done" to end the game.
```

Prompt template for limited inventory variant instances:

```
You are playing a text adventure game. I
will describe what you can perceive in
the game. You write the single action you
want to take in the game starting with >.
Only reply with an action.
For example:
> open cupboard
You can have up to two objects in your
inventory at the same time.

Your goal for this game is: $GOAL$

Once you have achieved your goal, write
"> done" to end the game.
```

Prompt template for synthetic word variant instances:

```
You are playing a text adventure game. I
will describe what you can perceive in
the game. You write the single action you
want to take in the game starting with >.
Only reply with an action.
For example:
> open cupboard
```

Action	Targets	Description	Epistemic	Pragmatic	Replaceable
open	"container" entities	Changes state of closed container entity to open	Yes	Yes	Yes
close	"container" entities	Changes state of open container entity to closed	No	Yes	Yes
take	"takeable" entities	Removes in/on state for "takeable" entity and adds in(entity,inventory) fact	No	Yes	Yes
put	"takeable"& "container"/"support" entities	Removes in(entity,inventory) state for "takeable" entity and adds in/on(entity,target) fact	No	Yes	Yes
go	"room"	Changes at state of player entity and all entities in inventory to target room	Yes	Yes	No
done	-	Ends the episode	No	Yes	No
examine	entities	Results in entity state feedback	Yes	No	No
look	'room'	Results in current room description feedback	No	No	No

Table 5: Basic action types used in AdentureGame. Targets are those for which the world state holds a fact assigning the listed state. Replaceable denotes actions that can be replaced with synthetic words in the *easy* and *medium* synthetic words experiments.

```
$NEW_WORDS_EXPLANATIONS$

Your goal for this game is: $GOAL$

Once you have achieved your goal, write
"> done" to end the game.
```

A description of the room the player starts in is appended to complete the initial prompts.
The placeholder `$GOAL$` in the templates is replaced with the task goal.

Example "easy" difficulty task goal: `Put the pillow on the table, the book on the table and the plate on the table.`

Example "hard" difficulty task goal: `Put the pillow on the counter, the book on the shelf and the plate on the table.`
For the synthetic words experiments, `$NEW_WORDS_EXPLANATIONS$` is replaced with `In addition to common actions, you can` followed by a synthetic action word and its explanation for *easy*, or a list of synthetic actions words for *medium* and *hard* instances.

E Environment Graphs

To illustrate differences between "easy"/"hard" environment and task complexity, Figures 12 and 13 show graph representations of initial game world states and task targets. House-shaped nodes are rooms, with arrow edges showing bidirectional connections between them. Round nodes are "movable" entities, connected to rectangular receptacles and rooms by edges labelled with their prepositional state. Dashed edges connect the movable task objects to their target receptacles and are labelled with the target prepositional state.

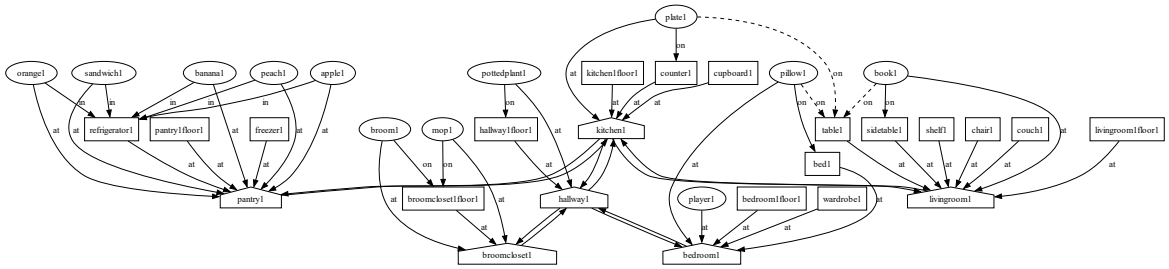

Figure 12: Graph representation of an “easy” basic instance.

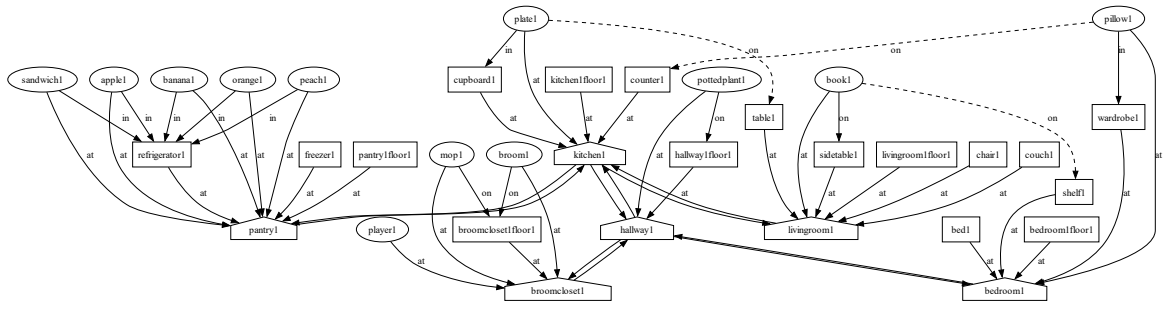

Figure 13: Graph representation of a “hard” basic instance.