`clembench`: Using Game Play to Evaluate Chat-Optimized Language Models as Conversational Agents

Kranti Chalamalasetti, Jana Götze, Sherzod Hakimov
Brielen Madureira, Philipp Sadler, David Schlangen [*]
Computational Linguistics, Department of Linguistics
University of Potsdam, Germany
first.last@uni-potsdam.de

Abstract

Recent work has proposed a methodology for the systematic evaluation of "Situated Language Understanding Agents"—agents that operate in rich linguistic and non-linguistic contexts—through testing them in carefully constructed interactive settings. Other recent work has argued that Large Language Models (LLMs), if suitably set up, can be understood as (simulators of) such agents. A connection suggests itself, which this paper explores: Can LLMs be evaluated meaningfully by exposing them to constrained game-like settings that are built to challenge specific capabilities? As a proof of concept, this paper investigates five interaction settings, showing that current chat-optimised LLMs are, to an extent, capable to follow game-play instructions. Both this capability and the quality of the game play, measured by how well the objectives of the different games are met, follows the development cycle, with newer models performing better. The metrics even for the comparatively simple example games are far from being saturated, suggesting that the proposed instrument will remain to have diagnostic value. Our general framework for implementing and evaluating games with LLMs is available at `https://github.com/clp-research/clembench`.

1 Introduction

There is an old joke:

> *A guy has a dog that plays checkers. "My goodness," everyone says, "that's amazing. What a brilliant dog!" — "Not really," he replies, "I beat him four games out of five."*

This joke nicely reflects where we are with interaction-tuned language models such as ChatGPT and GPT-4 (OpenAI, 2023).[1,2] While the public discussion is dominated by what amounts to an unguided breadth-first search of tasks that can be "done" by these models (seeing "sparks" of generality in the process, (Bubeck et al., 2023)), systematic investigations into how well these tasks are actually done, when looked at in depth, are only now beginning to appear (Liu et al., 2023; Bang et al., 2023)— often with results not dissimilar to what disappoints the dog owner in the joke, who apparently is looking for a challenging checkers partner and not a clever dog.

In this paper, we take the analogy even further and indeed look at how well these models can play interactive, language-based games, like that illustrated in Figure 1. In recent work, Schlangen

#	A	Game Master	B
1		`[A⟨GM] The word you need to describe is "expedition". The taboo words are "expedition", "journey", "discovery", "exploration".`	
2	`[A⟩GM] CLUE: A trip taken for a specific purpose.`		
3		`[GM|GM] [valid]`	
4		`[GM⟩B] CLUE: A trip taken for a specific purpose`	
5			`[GM⟨B] GUESS: Journey`
6		`[GM|GM] [valid, wrong]`	
7		`[A⟨GM] GUESS: journey`	
8	`[A⟩GM] CLUE: A planned and organized trip with a specific goal in mind.`		
9		`[GM|GM] [valid]`	
10		`[GM⟩B] CLUE: A planned and organized trip with a specific goal in mind`	
11			`[GM⟨B] GUESS: expedition`
12		`[GM|GM] [correct]`	

Figure 1: An episode of the `taboo` word game

---

[*] Contributions: KC coded and analysed the `wordle` games. JG did so for `taboo`, and managed the code repository. SH coded and analysed `drawing` and `reference`, implemented the LLM interfaces, and edited the appendix. BM coded and analysed `private/shared`, and organised the evaluation computations. PS coded `taboo` and implemented the main framework. DS conceived the original idea, managed the project, and edited the main part of the paper.

[1] Thanks are due to Carl T. Bergstrom for bringing this joke and its applicability to the situation to our attention; `https://fediscience.org/@ct_bergstrom/110273442253894015`.

[2] We will call such models cLLMs from here on, for "chat-optimized LLM", with the suggested pronunciation "clem".

(2023a) has argued that such *Dialogue Games* ("constructed activities driven by language use") are a good systematic way of probing for the situated language understanding of language-using agents. In other recent work, Andreas (2022) has argued that LLMs are models of such agents. We bring these claims together and investigate what we can learn about the capabilities of cLLMs by exposing them to constrained game-like settings. Beyond making it possible to control the buildup of context in which to interpret the language, the game setting also has the advantage that we can generate novel instances that are unlikely to have been seen in any kind of training data, even if the game itself may have been. We describe a framework for implementing such games in a way that they can be tested in self-play of cLLMs—through the use of a programmatic "Game Master" that controls the game flow, as in the example in Figure 1—and we show results for five games that we have implemented in this framework, testing as game play agents the models Anthropic Claude, AlephAlpha Luminous, GPT3, GPT3.5, GPT4.[3]

Our main findings are:

- Game instruction following generally is good, and is what marks the difference between models such as GPT-3 and newer models; likely as an effect of *instruction tuning* (Wei et al., 2022; Zhong et al., 2021) and learning from human feedback (Ouyang et al., 2022; Stiennon et al., 2020);
- The performance differences across games tracks the development cycle, with newer models generally performing better;
- The performance metrics are not saturated; and under the reasonable assumption that human performance would be near the ceiling, there is a wide gap between model performance and this.

Our contributions are:

- A flexible, extensible framework for the implementation of Dialogue Games as test instruments, which enables fast evaluation on a large (and extensible) set of models. This is available at `https://github.com/clp-research/clembench`.
- A collection of implemented and well-motivated games, together constituting version 1.0 of what we call the *clem benchmark*.

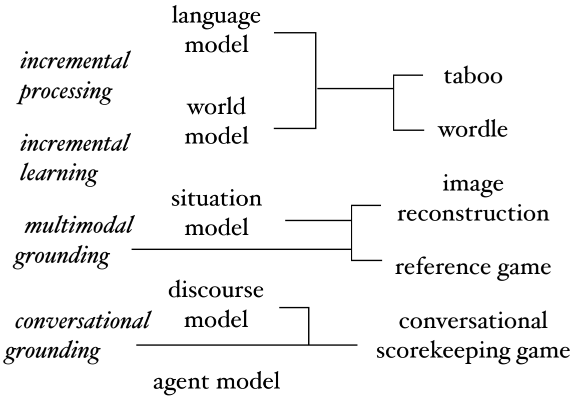


Figure 2: Anchoring Processes and Representational Domains from (Schlangen, 2023b) (left), and links to Dialogue Games described here

- An in-depth evaluation of the performance of current state-of-the-art cLLMs on these games.

2 Background: Situated Agents, Dialogue Games, and LLMs as Agent Models

Schlangen (2023a) introduces Dialogue Games as follows:

> A *Dialogue Game* is a constructed activity with a clear beginning and end, in which *players* attempt to reach a predetermined *goal state* primarily by means of producing and understanding linguistic material.

He argues that such Dialogue Games can serve as valid instruments for evaluating models of situated language understanding, provided that an argument can be given for how a specific game challenges aspects of the underlying construct. As a model of this (not directly observable, but to be measured) construct he takes his own proposal (Schlangen, 2023b), illustrated here in Figure 2, which analyses *situated language understanding* into a number of representational and procedural demands. Rather than going through these in detail here, we will illustrate them through the discussion of how the implemented games challenge these various aspects.

Andreas (2022) argues that LLMs "infer approximate, partial representations of the *beliefs, desires*, and *intentions* possessed by the agent that produced the context". If that is so, and if the finer-grained analysis of the relevant beliefs, desires, and intentions involved in game play that we reviewed in the previous paragraph is on the right track, then such

---

[3] gpt4: (OpenAI, 2023); gpt3.5: (Ouyang et al., 2022); gpt3: (Brown et al., 2020); claude: (Bai et al., 2022); luminous-supreme: (AlephAlpha, 2023)

games should form a valid instrument for measuring the degree to which LLMs do indeed approximate these capabilities.

Figure 2 illustrates how the example games implemented and evaluated here connect to the construct. (All games require a minimal form of discourse model being built, insofar as earlier information constrains later moves; and all games require a minimal type of agent model, insofar as the game instructions need to be taken on as own "intentions".) We will argue for these connections in detail below, but first we need to describe the scaffolding required to turn LLMs into game players.

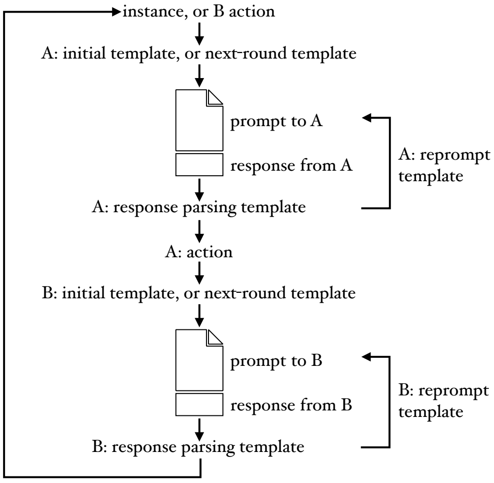

instance, or B action
A: initial template, or next-round template
prompt to A
response from A
A: reprompt template
A: response parsing template
A: action
B: initial template, or next-round template
prompt to B
response from B
B: reprompt template
B: response parsing template

Figure 3: Schematic View of the Game Flow

3 From Game To Benchmark

3.1 Terminology

First, some terminology: A *Dialogue Game Realisation* (DGR) fixes for a given game the prompt templates (with which the game is described to the players) and the logic of the *Game Master* (the programmatic component keeping the game on track; see below). An *instance* of a DGR fixes the goal (e.g., in a word-guessing game, the word to guess). A *data set* is a collection of instances. An *experiment* fixes the players that get to play through a data set; e.g., as either being a human participant, or as a computer model (with all its free parameters fixed). For each *episode* (play of an instance), the experiment results in an *interaction record*. This record is what gets evaluated, both at a turn-by-turn level (progress made in the game) as well as for whether (or to what degree) the goal was reached. The *benchmark* then is a specific collection of datasets, and a *benchmark result* is the evaluation of (the interaction records of) a fixed combination of players over the benchmark.

3.2 Turn-Based Text Games via Prompting

Not all kinds of Dialogue Games in the sense of Schlangen (2023a) can be realised with LLMs as players. For now, the games need to be text-based (although we do realise games below that use character-encodings for image-like structures), and they need to be turn-based, so that each turn can be one prompting of a player to produce its move for this turn. We realise single-player games as well as two-player games. In order to keep the interaction focussed on the interactional task / the Dialogue Game, we insert a (programmatic) *Game Master* into the interaction, whose task it is to keep track of the game state and to parse the reactions by the player, ensuring that only game-relevant actions are passed on and that the rules of the game are followed. In the taboo game shown in Figure 1 for example, the Game Master checks that the description given by player A does indeed not contain the "taboo" words (see description of the game below), before passing the message on to player B. Thereby, the "general purpose" nature of any participating model is hidden, and it can be evaluated purely for its performance in the game.

The games considered here are self-contained in the sense that each game opens with a description of the game rules and instructions regarding the *form* of the response expected from a player; the game play consists in the player choosing the *content* of the move. This makes it possible to separately evaluate the ability to play the game (follow the instructions) and the level of expertise at playing it (e.g., by how fast or how well the goal has been reached in a given episode). Figure 3 shows a schematic view of how the Game Master controls a two-player game by making use of prompt templates that are filled in based on the current game state.

3.3 The `clemgame` Framework

We have implemented a Python framework that provides the general pattern (prompting, Game Master) described above, and takes care of the infrastructure of routing the player turns to the various model APIs (or, in the case of human players, to an appropriate interface). It is easily extensible to include new language-processing models (of type "string to string"; that is, models that can be prompted with a context and that return text). The framework also takes care of the separation of instance collections into datasets, and of running (with different model settings) the experiments constituting the benchmark. The framework is available at `https://github.com/clp-research/clembench`, where also current results of the benchmark (to be discussed below in Section 5) are shown. All games described in the next section are implemented in this framework.

4 The Games in v1.0 of the Benchmark

All games described here challenge the rule-following capabilities of the players. In all games, the game objectives and the rules, including formal constraints on the game moves, are described verbally to the player. What these instructions leave implicit are general strategic considerations of game play, such as that repetitions in a guessing game have no strategic value. The Game Master validates each player move according to the formal constraints, and if after a certain amount of reprompting still no valid move is produced, the game is aborted. We measure for all games the proportion of games that were aborted in this way, giving us for each player a measure of their general ability to follow rules.

In the following, we briefly describe each game in general terms and give for each game a *preferred metric* with which to quantify the players' level of competence of playing it (beyond just following the rules so as to avoid the game play being aborted). Note that these metrics typically evaluate the pair of players together and cannot make role-based distinctions. All further details, such as how we realised the game through prompts and how we instantiated the realisation into game instances, are collected in the Appendix. The games described here are those we selected for a first version of the benchmark, with the aim of breadth with respect to the model in Figure 2; we see as an advantage of the framework that it is easy to implement more games, and expect the benchmark to be extended through contributions from the community.

4.1 A Simple Word Game: Taboo

In this game, one player has to describe to another player a concept, without using the concept name and any of a given list of semantically related words. The task of the other player then is to guess this word. If the player guesses wrongly, the first player can attempt a different description, following the same constraints. To play this game, the players must be able to access information from the interface between what we called above the *language model* and the *world model*, to provide and understand concept descriptions. The additional constraints on word use challenge a player's ability to apply given rules to the given situation.

#	A	Game Master	B

1		`[A⟨GM] [. . . ]`
		`clue:display`	

2	`[A⟩GM] guess: screen`
	`explanation: A screen is a common display device, and the word has 5 letters.`		

3		`[A⟨GM] The word should have exactly 5 letters. Please try again.`	

4	`[A⟩GM] guess: panel`
	`explanation: A panel can be a type of display, and the word has 5 letters.`		

5		`[GM⟩B] [guess & expl. relayed]`	

6			`[GM⟨B] agreement: yes`
			`explanation: panel is a suitable 5-letter word related to display`

7		`[A⟨GM] [ agreement relayed ]`	

8	`[A⟩GM] guess: panel`
	`explanation: A panel can be a type of display, and the word has 5 letters.`		

9		`[GM|GM]`
		`[no-change]`	

10		`[A⟨GM] guess-feedback: p a n e l`	

11	`[A⟩GM] guess:level`
	`explanation: Based on the feedback, the last two letters are "el". "Level" is a 5-letter word that contains "el" at the end.`		

Figure 4: Excerpt of *wordle+clue+critic* game play (GPT4/GPT4)

We have seen an example of game play already above in Figure 1, which indicated that the role of the Game Master is to provide the target word to player A, and to validate the players' moves (A cannot use taboo words; B either guesses correctly or not). We allow three rounds of guessing. The *preferred metric* for `taboo` is *speed*, scored non-linearly as $\frac{100}{n}$ (with n as the number of required moves), or 0 (if $n > 3$). For Figure 1 hence this would yield 50.

4.2 Word-Guessing w/ Letter-Based Feedback

We also implemented some variations of the popular word-guessing game "Wordle".[4] The basic mechanics of this game is that letter-based feedback is provided on guesses of 5-letter words, which incrementally constrains the set of possible words.

---

[4] `https://www.nytimes.com/games/wordle`

If the target word for example is APPLE, a guess of ALONE would yield the following information: A appears at this position, L appears elsewhere, O does not occur, N does not occur, E occurs at this position. We also implement non-standard variations where a textual clue is given at the beginning as to the identity of the word. These games are one-player games (although we technically realised the computation of letter-feedback as the contribution of a player B). We also implemented a variant where there is a more active additional player, who can give feedback on the choice of player A before it is played, giving A the opportunity to select differently. These game variants again challenge knowledge from *language* and *world model*, as well as, in a rudimentary form, in the "critic" variant, simulating processes of conversational grounding / negotation. Figure 4 shows an excerpt of a game played with critic. The *preferred metric* for all variants again is *speed* (with a maximum of 6 guesses).

4.3 Drawing Instruction Giving and Following

In this game, player A is given an image (here represented as a grid of characters, with □ representing an empty cell), and their task is to instruct player B to reconstruct this image, starting from an empty grid. (See Figure 5 for an example.) Hence, to be successful both player A and B must form, in a limited multimodal way, a model of a (very much abstracted) situation. The game stops when player A signals that their description is complete. The *preferred metric* is the *F1-score* of player B's grid relative to player A's target and the non-empty "pixels": if all target cells have been changed as desired, it is 100, if none have, it is 0. We test with *compact* instances, which allow for higher level descriptions (as in the example), and *random* grids, which do not; see Appendix D.

#	A	Game Master	B
1		[A⟨GM] Ok. Please do this for the following example, which is a 5 by 5 grid. □□□□□ B B B B B □□□□□ B B B B B □□□□□ What is your next instruction?	
2	[A⟩GM] Instruction: Put B in all cells of the second and fourth rows.		
3		[GM|GM] [valid]	
4		[GM⟩B] [. . .] Instruction: Put B in all cells of the second and fourth rows.	
5			[GM⟨B] OUTPUT: □□□□□ B B B B B □□□□□ B B B B B B B B BB
6		[GM|GM] [valid]	
7		[A⟨GM] What is your next instruction?	
8	[A⟩GM] Instruction: DONE		

Figure 5: An episode of the `drawing` game

4.4 A Picture Reference Game

We also implemented a simple *Lewis Signalling Game* (Lewis, 1969), where A is presented with three grids (of the type also used in `drawing`; shown in Figure 6) and the task to make B (who is also presented with the same three grids, but potentially in a different order) identify a pre-specified one. As in `drawing`, this game challenges the formation of a *situation model*, and, to be done efficiently, needs access to analogical information from the agent's *world model* (e.g., to describe the second grid in Figure 6 as "looks like a T"). There is a long tradition in psychology to use such reference games to provide insights into communicative behaviour (see, e.g., (Yule, 1997)). The *preferred metric* for this game is a simple binary *success* measure: Did B identify the target, or not?

1st Grid	2nd Grid	3rd Grid
XXXXX	XXXXX	XXXXX
□□X□□	□□X□□	□□□□□
XXXXX	□□X□□	□□□□□
□□X□□	□□X□□	□□□□□
XXXXX	□□X□□	XXXXX

Figure 6: Sample grids for the `reference` game

4.5 Scorekeeping: Private and Shared

The final game in v1.0 is structurally a bit different, as besides the main dialogical interaction, there is a secondary *probing interaction* going on, which is the target of interest here. In the primary interaction, a *customer agent* goes through a form with a *customer*. In the probing interaction, the game master probes the customer agent on what they "think" that the agent already knows. Each instance specifies the list of information that is to be exchanged, and so the game master can keep track of which information has already been shared. If a piece of information has not yet been shared, the customer should not expect the agent to know it. This game challenges the keeping of an *agent model* and its update through *conversational grounding*

#	A	Game Master	B
1		[A⟨GM] [initial prompt]	
2			[GM⟨B] What class do you prefer?
3	[A⟩GM] ANSWER: Economy.		
4			[GM⟨B] Which means of transportation do you prefer?
5	[A⟩GM] ANSWER: Train.		
6			[GM⟨B] Please inform your destination.
7	[A⟩GM] ANSWER: Stuttgart.		
8			[GM⟨B] What is the origin of your trip?
9	[A⟩GM] ANSWER: London.		
10			[GM⟨B] When is the trip?
11	[A⟩GM] ANSWER: In May.		

Figure 7: An example of the primary interaction in `private/shared`

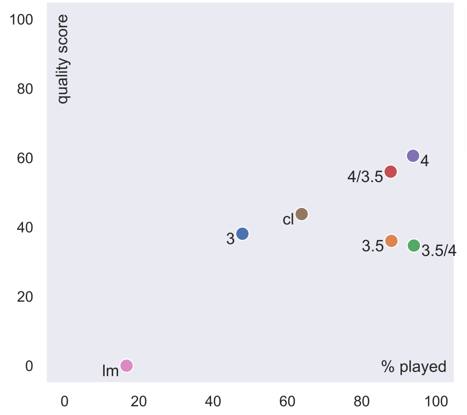


Figure 9: Overview of main results

processes. Figure 7 shows an edited transcript of the primary interaction, and Figure 8 shows an example of a probing interaction that the game master can interject. For evaluation, we compute the slot-filling accuracy throughout the main interaction and the agreement between the model's answers and the ground truth in the probing rounds. Because each probe is a binary decision (shared or not), the random performance would be high, so we use Cohen's Kappa (Cohen, 1960) to control for chance. The *preferred metric* is the harmonic mean between the slot-filling accuracy and the probing Kappa (truncated at 0).

#	A	Game Master	B
1		[A⟨GM] ME: Do you think the travel agent knows where you depart from? Please answer yes or no.	
2	[A⟩GM] ASIDE: No.		
3		[A⟨GM] ME: Do you believe that the travel agent is aware of where you are going? Please answer yes or no.	
4	[A⟩GM] ASIDE: Yes.		
5		[A⟨GM] ME: Does the travel agent know how you wish to travel? Please answer yes or no.	
6	[A⟩GM] ASIDE: No.		
7		[A⟨GM] ME: Have you informed the travel agent about your class preference? Please answer yes or no.	
8	[A⟩GM] ASIDE: Yes.		
9		[A⟨GM] ME: Is the travel agent aware of the dates of your trip? Please answer yes or no.	
10	[A⟩GM] ASIDE: No.		

Figure 8: An example of the secondary interaction in `private/shared`

5 Results

As detailed in the Appendix, the full benchmark (v1.0) consists of 214 instances: 30 for `taboo`, 30 for `wordle`, 30 for `wordle+clue`, 30 for `wordle+clue+critic`, 40 for `drawing`, 36 for `reference`, and 20 for `private/shared`.

We ran the benchmark on the models shown in Table 2 with self-play (a model plays all players in a game). In addition, we run pairs of `gpt-4` and `gpt-3.5` to test if a supposedly better model (here `gpt-4`) can leverage the other. Following Srivastava et al. (2022), we requested greedy sampling (i.e., temperature 0). One run of the benchmark, somewhat surprisingly, on average took more than 600 minutes to complete, due to API latency,[5] and cost around 50$ in API fees.

Table 1 gives the overall results of this benchmark run. For each model (pairing), we first show what percentage of instances were played to completion (i.e., not aborted because of problems of the players in following the instructions). We then show what the quality of the play was for those played instances, using each game's preferred metric. The first column (macro-)averages the numbers over all games, with the remaining ones giving the per-game results. Figure 9 provides the same information in a graphical format, plotting "percentage played" against "quality". A perfect model—and, we suspect, since these are simple games, human performance—would be clustered in the top right corner (all instances played, with high quality). As we can see from the results, the GPT family tends to perform better than the other models we tested, with an increase in quality from 3 to 3.5 to 4. There is a jump in the ability to play games to completion (that is, to follow the prompt instructions as to the *format* of the game play moves)

---

[5] There is a high variance between games. Also, this is accumulated time, although benchmark runs on different models could in principle be parallelised.

model	game:	`all`	`drawing`	`priv/sh`	`reference`	`taboo`	`wordle`	`wordle+cl`	`wordle+cl+cr`
lm/lm	% played	16.67	0.0	0.0	0.0	0.0	100.0	6.67	10.0
	qlty score	0.0	/	/	/	/	0.0 (0.0)	0.0 (0.0)	0.0 (0.0)
cl/cl	% played	63.81	0.0	100.0	**100.0**	0.0	100.0	96.67	50.0
	qlty score	43.77	/	60.27 (24.65)	**88.89** (31.87)	/	0.0 (0.0)	40.8 (46.42)	28.89 (36.4)
3/3	% played	47.86	65.0	10.0	83.33	23.33	73.33	46.67	33.33
	qlty score	38.09	44.77 (25.59)	55.32 (19.65)	63.33 (49.01)	14.29 (37.8)	0.0 (0.0)	46.43 (41.44)	42.5 (50.07)
3.5/3.5	% played	87.98	97.5	85.0	100.0	56.67	100.0	90.0	86.67
	qlty score	36.02	65.41 (25.32)	56.72 (27.75)	66.67 (47.81)	29.41 (46.97)	0.0 (0.0)	18.52 (39.58)	15.38 (26.21)
3.5/4	% played	**94.03**	**97.5**	/	100.0	**86.67**	100.0	90.0	90.0
	qlty score	34.68	70.28 (24.08)	/	72.22 (45.43)	28.85 (40.43)	0.0 (0.0)	18.52 (39.58)	18.21 (29.51)
4/3.5	% played	87.78	80.0	/	100.0	46.67	100.0	100.0	100.0
	qlty score	56.01	79.47 (23.93)	/	61.11 (49.44)	**96.43** (13.36)	3.56 (9.55)	47.06 (42.27)	48.44 (45.27))
4/4	% played	93.81	80.0	**100.0** (0.0)	100.0	76.67	100.0	100.0	100.0
	qlty score	**60.59**	**92.09** (12.04)	**78.55** (9.74)	77.78 (42.16)	73.19 (37.18)	**4.56** (10.59)	**47.89** (41.55)	**50.11** (44.98)

Table 1: Results Overview. For each model (pairing), shows how many games were played to completion (% played), an indicator of rule-following capabilities. "qlty score" indicates how well the completed games were played (higher is better, max is 100). `all` is the average over all games scores, the remaining columns show results broken down by game (averaged over all episodes).

Model	Version	Here	P	T	I
`gpt-4`	`0314`	4	n/a	n/a	Y
`gpt-3.5-turbo`	`0301`	3.5	n/a	n/a	Y
`text-davinci`	`003`	3	175	300	Y
`claude`	`v1.3`	cl	52	n/a	Y
`luminous-supreme`	`2023-01`	lm	70	588	Y

Table 2: The evaluated models with the details about number of parameters in billions (P), trained data size in billions (T), and whether they were instruction tuned (I). Y: yes, n/a: not available publicly.

from 3 to 3.5, with a smaller increase from 3.5 to 4. Still, even the best performing model, GPT-4, does not reach 100% on "percentage played", with the reduction mostly due to `drawing` and, somewhat surprisingly, `taboo` — perhaps due to the *negative* nature of the game constraints ("don't say X").

When it comes to the quality of the game play (in those episodes played to completion), we see a similar trend, with GPT4 overall performing best. We also see that there is ample room for improvement, with the best average score standing at 60.59. An outlier in terms of quality is `wordle`, where even though almost all models manage to stick to the local rules (produce a 5-letter word), even the best-performing model, GPT4, only reaches 4.56 on the quality metric, indicating that very few games are actually solved, and those only at the last attempt. This indicates that all models fail at integrating the feedback across turns and using it to constrain their guesses. The capabilities of dealing with verbal meaning definitions are shown by the large improvement that `wordle+clue` exhibits (to 47.89). Interestingly, GPT4 is able to profit from (via another instance, self-)criticism, improving further to 50.11.

Again somewhat surprisingly, performance on the "multimodal" games (which require verbalisation of character-based graphics) is not bad. For `drawing`, as a deviation from the trend, GPT3.5 proved to be better at sticking to the game format (97.5% of episodes played to completion), although GPT4 still reached higher quality on its completed games. `reference` sees Claude performing best, against the trend for all other games.

6 Related Work

Playing games and learning from self-play stands at the beginnings of the "deep learning revolution" (Mnih et al., 2013; Silver et al., 2017). What is different here is the zero- or few-shot nature of our test, where the testing mode is different from the learning mode—this of course only being enabled by "foundation models" (Brown et al., 2020). The latest—apparent—qualitative jump has only recently been taken, so there are not that many papers yet that attempt a systematic evaluation; see, *inter alia*, (Liu et al., 2023; Bang et al., 2023). To our knowledge, game play of the kind proposed here has not yet been used for the systematic evaluation of these models. The idea of testing game play is mentioned in (Bang et al., 2023; Bubeck et al., 2023), and also already technically possible in (Srivastava et al., 2022), but has not been systematically executed there.

A superficial similarity also exists to approaches like HuggingGPT (Shen et al., 2023) in that these approaches pair LLMs with scaffolding (as in our Game Master). A crucial difference, however, is that for us the task of the Game Master is to *constrain* the LLM and to “keep it focused”, as it were, on the game, rather than to *extend* its capabilities.

Park et al. (2023) also acted on the realisation that cLLMs can simulate agents which can be put into “self-play”, but developed this idea in a different direction, towards investigating the emerging “social behaviour”.

7 Roadmap

Important steps on our roadmap include testing the models’ abilities to handle languages other than English and integrating the framework with the slurk chat tool (Götze et al., 2022) in order to enable game plays in which one or more of the players is a person. We also plan to experiment with games that have more than two players as well as games that require multimodal context such as images.

8 Conclusions

We have shown that current chat-optimised large language models can indeed serve as models of interactive agents, at least for controlled and rule-constituted activities such as verbal games. We have described our general implementation of a framework for implementing rules to be played in “self-play” by such models, with the main idea being that a programmatic component, the “Game Master” can control the interaction and ensure that only formally correct moves are registered. We have described our example implementations and instantiations of such games, arguing that the span the breadth of the sub-capabilities involved in situated language processing (if only on a relatively superficial level). Finally, we have shown that the evaluation of the game play can serve as an instrument to distinguish between models in terms of their language capabilities. With this work, we have aimed to open a complimentary avenue for evaluating these models, beyond more classical NLP tasks, and into the realm of interactive language use. Much remains to be done, but we hope that our framework can support some of this future work.

Limitations

Limits on reproducibility

The models under evaluation are only accessible via programming interfaces which basically add a black box on top of a black box. The mechanics (and exact models invoked) behind these interfaces might change at any time and consequently the results of successive runs might vary arbitrarily. For the closed models tested here, the best we can do is to provide the timestamp of the testing and the versioning information, to the extent that it is available to us.

Ethics Statement

Using paid proprietary APIs with underlying models about which little is known (training data, model architecture) in academic research is less than ideal. At the moment, the models tested here seem to be the only ones that are even able to follow the structure of the games as instructed. It is our hope that open models will catch up soon, and proper research can be done with them.

References

AlephAlpha. 2023. What is luminous? `https://docs.aleph-alpha.com/docs/introduction/luminous/`. Accessed: 2023-05-19.

Jacob Andreas. 2022. Language models as agent models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5769–5779, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom B. Brown, Jack Clark, Sam McCandlish, Chris Olah, Benjamin Mann, and Jared Kaplan. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *CoRR*, abs/2204.05862.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity.

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram Version 1.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 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*.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4.

Herbert H Clark and Susan E Brennan. 1991. Grounding in communication. In *Perspectives on socially shared cognition.*, pages 127–149. American Psychological Association.

Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.

Jana Götze, Maike Paetzel-Prüsmann, Wencke Liermann, Tim Diekmann, and David Schlangen. 2022. The slurk interaction server framework: Better data for better dialog models. In *Proceedings of the Language Resources and Evaluation Conference*, pages 4069–4078, Marseille, France. European Language Resources Association.

David Lewis. 1969. *Convention*. Harvard University Press.

David Lewis. 1979. Scorekeeping in a language game. In *Semantics from different points of view*, pages 172–187. Springer.

Nelson F. Liu, Tianyi Zhang, and Percy Liang. 2023. Evaluating verifiability in generative search engines.

Brielen Madureira and David Schlangen. 2022. Can visual dialogue models do scorekeeping? exploring how dialogue representations incrementally encode shared knowledge. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 651–664, Dublin, Ireland. Association for Computational Linguistics.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. 2013. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Gray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. 2023. Generative agents: Interactive simulacra of human behavior.

David Schlangen. 2023a. Dialogue games for benchmarking language understanding: Motivation, taxonomy, strategy. *CoRR*, abs/2304.07007.

David Schlangen. 2023b. What A Situated Language-Using Agent Must be Able to Do: A Top-Down Analysis. *CoRR*, abs/2302.08590.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2023. Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George Van Den Driessche, Thore Graepel, and Demis Hassabis. 2017. Mastering the game of Go without human knowledge. *Nature*, 550(7676):354–359.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R. Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, Agnieszka Kluska, Aitor Lewkowycz, Akshat Agarwal, Alethea Power, Alex Ray, Alex Warstadt, Alexander W. Kocurek, Ali Safaya, Ali Tazarv, Alice Xiang, Alicia Parrish, Allen Nie, Aman Hussain, Amanda Askell, Amanda Dsouza, Ameet Rahane, Anantharaman S. Iyer, Anders Andreassen, Andrea Santilli, Andreas Stuhlmüller, Andrew M. Dai, Andrew La, Andrew K. Lampinen, Andy Zou, Angela Jiang, Angelica Chen, Anh Vuong, Animesh Gupta, Anna Gottardi, Antonio Norelli, Anu Venkatesh, Arash Gholamidavoodi, Arfa Tabassum, Arul Menezes, Arun Kirubarajan, Asher Mullokandov, Ashish Sabharwal, Austin Herrick, Avia Efrat, Aykut Erdem, Ayla Karakas, and et al. 2022. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *CoRR*, abs/2206.04615.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pages 3008–3021. Curran Associates, Inc.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

George Yule. 1997. *Referential Communication Tasks*. Routledge, New York, USA.

Ruiqi Zhong, Kristy Lee, Zheng Zhang, and Dan Klein. 2021. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2856–2878, Punta Cana, Dominican Republic. Association for Computational Linguistics.

A Common metrics

Besides each game's specific scores, the following metrics are computed for all games:

- **Preferred Score**: A custom performance score, normalised to the interval $[0, 100]$, used to compare models across different games.
- **Aborted**: At the episode level, either 0 or 1 whether the game play has been aborted (1) or not (0). A game counts as aborted when a violation of the game rules happens, for example a response is not parsable by the rule that specifies it's format as "*TYPE: <text>*" (or re-prompt for n turns). This metric does not include games that are succeeded or lost. *Measures: episode performance*.
- **Lose**: At the episode level, either 0 or 1 whether the game play has been successful (0) or not (1) (this ratio does not include aborted games; the game is lost, when the game goal is not reached within the declared number of max_turns, in this sense it's the opposite of success). *Measures: episode performance*.
- **Success**: At the episode level, either 0 or 1 whether the game play has been successful (1) or not (0) (this ratio does not include aborted games; the game is successful, when the game goal is reached within the declared number of max_turns, in this sense it's the opposite of lost). *Measures: episode performance*.
- **Request Count**: total number of request given to the model by the GM (usually 1 per turn, but for games with re-prompting this might be >1 per turn). *Measured at: turn and episode level.*
- **Parsed Request Count**: total number of request that could be parsed successfully (the model's response complies to the game rules; accumulates over the episode). *Measured at: turn and episode level.*
- **Violated Request Count**: game master checks the outputted text and decides whether it matches the "game form" (also as a log action), if not then this is a violation of the game rules; total count of failures in a episode; turn-based (can be >= 0). *Measured at: turn and episode level.*
- **Request Success Ratio**: parsing success rate - or prompt has been successful if the output can be parsed properly (PARSED REQUEST COUNT/REQUEST COUNT). *Measures: episode performance*.

B Taboo Game Details

In this game a *Describer* describes a target word for a *Guesser*. The Describer must explain the target word concept without using neither the word itself, nor a number of related words. For example, when the target word is *flashlight*, the Describer cannot use the words *light or flash*. After each incorrect guess by the Guesser, the Describer can add to their description. The game ends when the Guesser guesses correctly or a maximum number of turns has been reached.

The game tests a cLLM's ability to describe concepts and give meaning definitions. It also tests its helpfulness in the game context, e.g. if a Describer does not alter or extend its initial description after an incorrect guess, we consider this unhelpful behavior. Similarly, if a Guesser repeats an earlier guess, it has not understood the game goal well enough to make real progress at each turn.

Instantiation We instantiate this game by setting the maximum number of guesses to 3 and we use target words that vary according to their frequency.

We use an English word frequency list based on web data (Brants and Franz, 2006)[6] to derive a list of lemmatized content words from which we choose words from 3 levels of frequency by dividing the word list into 3 equally-sized bins after removing words with a frequency of less than 5 per 1 million tokens. The remaining low-frequency words occur up 9.4 times per 1 million tokens, the medium-frequency words occur up to 25.1 times per 1 million tokens and the high-frequency tokens occur up to 12951.3 times in 1 million tokens. After a random selection from each frequency level, we manually ensure that the final word list does not contain inappropriate words such as vulgar language or proper names. The final word lists contain 10 words each. We use word frequency as a proxy for the difficulty of a game instance. The players are each given their own prompts, as shown in Figure 10.

---

[6] https://www.kaggle.com/datasets/rtatman/english-word-frequency

TEMPLATE B.0.1
You are playing a collaborative word guessing game in which you have to describe a target word for another player to guess.

Rules:
(a) You have to reply in the form: CLUE: <some text>. Guesses from the other player will start with GUESS.
(b) You cannot use the target word itself, parts or morphological variants of it in your description.
(c) In addition, the same rules apply for related words which are provided below.

End conditions:
(i) If you use the target word or a related word in your description, then you lose.
(ii) If the other player can guess the target word in N tries, you both win.

Let us start.

This is the target word that you need to describe and that the other player needs to guess:

`$TARGET_WORD$`

Related words are:

`$REL_WORD$`

Important: You are under time pressure, give short descriptions that are to the point!

TEMPLATE B.0.2
You are playing a collaborative word guessing game in which you have to guess a target word that another player describes to you.

You can make one guess at each trial. You win when you guess the target word. You lose when you cannot guess it in `$N$` tries.

After each trial you will get a new hint from the other player which starts with CLUE.

Make your guesses by just saying the word using the following form: GUESS: <a word>

Let us start.

Figure 10: The Describer and Guesser prompts for the Taboo game.

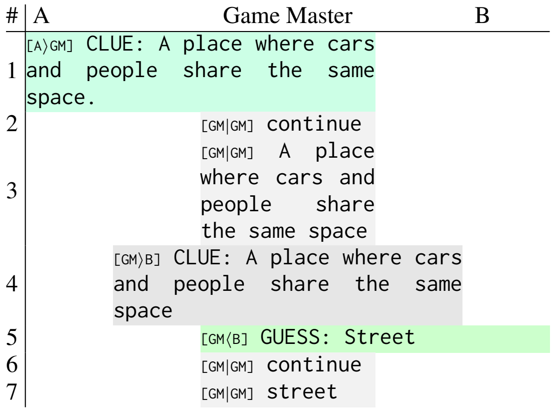
model	% played ↑	o/w success ↑	speed ↑
3–3	23	13	14.29
3.5–3.5	57	29	29.41
3.5–4	87	37	28.85
4–3.5	47	100	96.43
4–4	77	87	73.19
cl–cl	0	/	/
lm–lm	0	/	/

Table 3: A closer look at the models' performances for taboo: games played and the percentage of those ended successfully.

Evaluation We measure the following metrics at the episode-level:

1. **Success**: Whether or not the Guesser guessed the target word. In the example, success is 0.
2. **Abort**: 1 if any player did not follow the rules, and 0 otherwise. In the example, abort is 1.
3. **Speed**: How early the Guesser guessed the word as measured by $100/t$, where t is the turn number in which the target was found. When the game was unsuccessful or aborted, as in the example, speed is undefined.

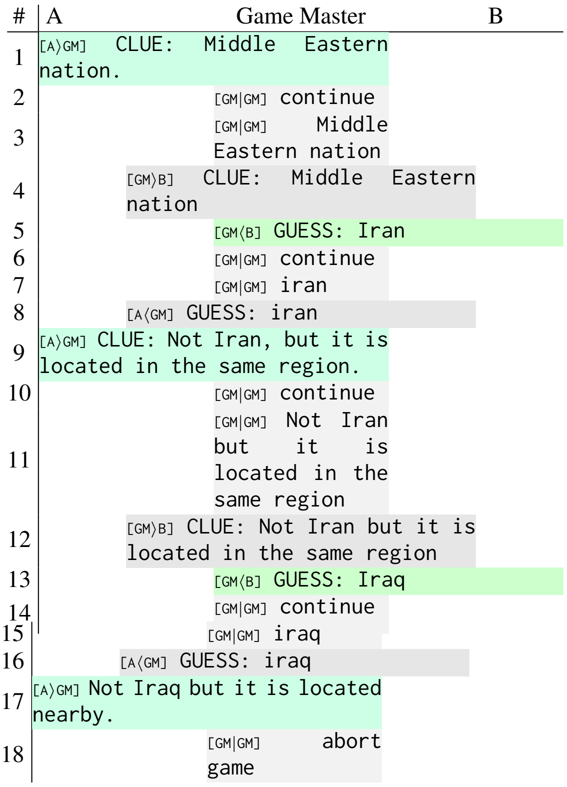
Example transcripts We present example transcripts in Figures 11 and 12.



#	A	Game Master	B
1	[A⟩GM] CLUE: A place where cars and people share the same space.		
2		[GM|GM] continue	
3		[GM|GM] A place where cars and people share the same space	
4			[GM⟩B] CLUE: A place where cars and people share the same space
5			[GM⟨B] GUESS: Street
6		[GM|GM] continue	
7		[GM|GM] street	

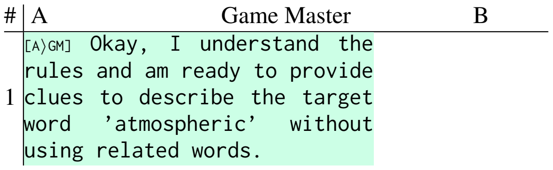
Figure 11: Example of a successful gameplay. The target word was *street*, taboo words were *road, asphalt, drive*.

Results The results in Table 1 indicate that Claude and Luminous are not able to play the game at all. The first response of these models is an acknowledgement of the game rules instead of an



#	A	Game Master	B
1	[A⟩GM] CLUE: Middle Eastern nation.		
2		[GM|GM] continue	
3		[GM|GM] Middle Eastern nation	
4			[GM⟩B] CLUE: Middle Eastern nation
5			[GM⟨B] GUESS: Iran
6		[GM|GM] continue	
7		[GM|GM] iran	
8	[A⟨GM] GUESS: iran		
9	[A⟩GM] CLUE: Not Iran, but it is located in the same region.		
10		[GM|GM] continue	
11		[GM|GM] Not Iran but it is located in the same region	
12			[GM⟩B] CLUE: Not Iran but it is located in the same region
13			[GM⟨B] GUESS: Iraq
14		[GM|GM] continue	
15		[GM|GM] iraq	
16	[A⟨GM] GUESS: iraq		
17	[A⟩GM] Not Iraq but it is located nearby.		
18		[GM|GM] abort game	

Figure 12: Example of a game that was aborted because the Describer violated the game rules (the description has to begin with `CLUE:`). The target word was *israel*, taboo words were *country, tel aviv, jew*.



#	A	Game Master	B
1	[A⟩GM] Okay, I understand the rules and am ready to provide clues to describe the target word 'atmospheric' without using related words.		

Figure 13: Claude fails to immediately produce the initial description for the guesser. Similar behaviour is seen with Luminous-supreme.

initial description (Figure 13), i.e. they fail to comply to the initial game rule that constrains them in what is allowed to be produced.

The GPT-based models are able to follow these rules and play the game in 23-87 percent of the time. We see that an increase in played games goes along with the development of the model (3→23; 3.5→57; 4→77). In particular, we see that the more recent chat-optimized models are able to succeed more often (3→13; 3.5→29; 4→87) when they are playing the game. The instruction finetuning seems critical to obey to the games rules (meaning to constrain the outputs to follow a specific pattern).

Pairing up different models shows that with gpt-4 as a guesser more games are actually played (not aborted). This indicates that gpt-3.5 might be a worse guesser than gpt-4 (less games played and less successful completions). Furthermore, when gpt-4 is playing as describer, we see an improvement of the gpt-3.5 guesser to perfect play and high speed. (however, only about half of the games are played) which indicates that gpt-4 represents a strong language and world model and is a useful description provider for gpt-3.5. Still, especially the number of games played (without a rule violation) is less than what we would expect from a human player. We will test human abilities to play this game in a future iteration using `slurk`.

C Wordle Game Details

The popular word guessing game “Wordle” gained global attention, in which players are challenged to guess a five-letter word in six attempts. After each guess, the player receives feedback indicating which letters are in the correct position, which letters are correct but in the wrong position, and which letters are incorrect to help them strategise their next guess. The objective of the game is to guess the target word using the fewest possible guesses, and the game ends when the player guesses correctly or exhausts all six attempts.

C.1 Wordle (Traditional Variant)

This game evaluates three key aspects of cLLM’s capabilities. Firstly, it assesses how well the cLLM comprehends the game rules, which involves generating valid English words consisting of exactly five letters. Secondly, it measures how effectively cLLM uses guess feedback to generate its next guesses. Thirdly, it measures how quickly cLLM can guess the target word if it succeeds.

In traditional gameplay, cLLM plays the role of “Player A”, and a deterministic wordle bot plays the role of “Player B”. Game begins with the game master prompting Player A to guess the target word. The game master parses Player A’s response and forwards it to Player B, which evaluates the closeness of the guess word to the target word and returns the feedback. The game master sends the feedback to Player A for the next guess and the cycle continues until the target word is guessed correctly or all six attempts are exhausted. The prompt template of this variant is available in Figure 15a.

C.2 Wordle (+ Semantics-Based Clue)

This is a Wordle variant, where the guesser (Player A) gets a clue before starting to guess. For example, for the target word PRIDE, the clue could be “pack of lions”. The rest of the game rules follow the same as the traditional game variant. cLLM plays the role of the “player A”, and a deterministic wordle bot plays the role of “player B”.

The primary aim of testing this variant is to evaluate the efficacy of Player A in effectively utilising the supplementary information provided by a clue to improve its guess of the target word. The clue serves as an aid to narrow down the possible word options. The success of the game depends on Player A’s ability to integrate the clue with the guess_feedback. Player A’s explanation offers insights into how the cLLM links the clue phrase and the guess_feedback. The prompt template is available in Figure 15b.

C.3 Wordle (+ Clue, + Critic)

This game variant also begins with the guesser (Player A) who attempts to guess the target word based on a given clue. In contrast to other game variants, where the guessed word is immediately evaluated for its proximity to the target word, in this variant, the guessed word and the clue are forwarded to another player known as the *critic*, to get an opinion on the correctness of the guess. The critic responds with either agreement or disagreement, providing their rationale based on the information given. The critic’s response is then relayed to the guesser, who can decide to stick with their initial guess or change it based on the feedback received. Figure 16a shows the prompt structure for the Player A, Figure 16b shows the prompt structure for the critic role and Figure 14 depicts the prompts fed to the guesser to share the critic's opinion.

This game variant helps to investigate the influence of the critic's role in the guesser's performance and can lead to interesting possibilities in human-machine interaction, where the human can be aided by the cLLM as the critic. We tested the game using the same cLLM for both roles, as well as different cLLMs for each role, employing distinct prompts for each.

C.4 Instantiation, Error Handling & Evaluation

Instantiation In our experiments, we use a list of 2,309 possible target words and a list of 12,953 valid guess words.[7] For textual clues, we use New York Times crossword clues.[8]. We sort the target words by word frequency.[9] Out of the initial 2,309 target words, frequency details are not available for one word, and clues are not available for 39 words. These words are subsequently excluded from the experiments. The remaining 2,269 target words are sorted based on their word frequency (descending frequency) and then divided into three equal groups. The first group which contains high-frequency words, has a total of 756 words. The second group, consisting of words with medium frequency, also contains 756 words. Finally, the third group, which contains low-frequency words, has a total of 757 words. To evaluate our methodology, we chose (random seed: 42) 10 words from each frequency group, resulting in a total of 30 target words for evaluation purposes, for each game variant. As metrics, we keep track of the success rate (how often the guesser guessed the target word, within the limit of 6 guesses), the average speed (if successful, then at which turn), and for each turn closeness (based on the letter-feedback). We also keep track of whether the guesser repeats a guess (a strategic failure), and, in the critic variant, whether the guesser changes the guess after feedback.

Error Handling The experiments revolve closely around the cLLM models, which are expected to respond in a specific format and adhere to certain rules. However, there are multiple scenarios where the responses from these models may result in errors.

1. In the Wordle game, a subset of valid five-letter English words is used. In certain scenarios, the guesser (Player A - cLLM) may guess a valid 5-letter word that is not among the allowed guesses. In such cases, cLLM will be asked to guess another word. This reprompting process continues until cLLM makes an allowed guess.
2. The Wordle game has a strict rule that allows guessing only 5-letter words. Sometimes, the models respond with words that do not adhere to this restriction, causing the reprompting. We allow two reprompting attempts, after which the game is considered aborted.
3. Sometimes, the response of the cLLM doesn't follow the expected format as stated in the prompt. In such cases, we reprompt the cLLM to generate the response in the expected format. When faced with these circumstances, we usually give two reprompts before declaring the game as aborted.

Evaluation For each episode, we record the number of guesses made by the guesser. If the guesser correctly guessed the word in six or fewer attempts, the game is counted as a success. If the guesser exhausted all six attempts, the game is counted as a failure. If the guesser's response does not conform to the game rules, the game is counted as aborted. Of the successful games, the average number of guesses taken to guess the word is computed. For all the games, we also measured how close the guess gets to the target word with each turn and how many times the guesser repeats the same guess. For the episodes, where the critic is available we count how many times the guesser changed the guess based on the critic's opinion. The following are the metrics measured for each episode.

1. **Success**: This is a binary value and measures whether the guesser guessed the target word or not.
2. **Aborted**: This is a binary value and measures whether the game aborted due to non-compliance with the game rules (words not containing 5 letters, words containing symbols other than alphabets).

---

[7] https://github.com/3b1b/videos/blob/master/_2022/wordle/data/allowed_words.txt https://github.com/3b1b/videos/blob/master/_2022/wordle/data/possible_words.txt

[8] https://www.kaggle.com/datasets/darinhawley/new-york-times-crossword-clues-answers-19932021

[9] https://www.kaggle.com/datasets/rtatman/english-word-frequency

	wordle			**wordle + clue**			**wordle + clue + critic**		
Model	Speed	Played	Success	Speed	Played	Success	Speed	Played	Success
3–3	0.0	0.73	0.0	46	0.46	0.30	42.5	0.33	0.16
3.5–3.5	0.0	1.0	0.0	18	0.9	0.16	15.3	0.86	0.26
3.5–4	0.0	1.0	0.0	18	0.9	0.16	18.2	0.90	0.33
4–3.5	3.5	1.0	0.13	47	1.0	0.66	48.4	1.0	0.63
4–4	4.5	1.0	0.16	47	1.0	**0.70**	**50.1**	**1.0**	0.63
cl–cl	0.0	1.0	0.0	40	0.96	0.46	28.8	0.50	0.23
lm–lm	0.0	1.0	0.0	0.0	0.06	0	0	0.10	0

Table 4: A closer look at the models' performances for wordle (traditional, clue, critic variants): games played and the percentage of those ended successfully.

3. **Speed**: This contains the score ranging from 0-to-100 and computes how quickly the guesser guessed the word.
4. **Closeness**: This contains the score ranging from 0-to-25 and determines how effectively the guesser utilizes the guess feedback. If a letter is at the correct position 5-points are awarded, and 3-points for letter at other position and 0-points for incorrect letters, leading to 25 points for a correct guess. Ideally this score should be improved across the turns.
5. **Repetition-Guesser**: This is a numeric value and assess how often the guesser repeated a guess.
6. **Change-Of-Opinion-Guesser**: This is a numeric value and calculates the number of times guesser changing/retaining the guess,

Results The detailed results for all three variants of the wordle game is given in Table 4. The combination of *clue* + *critic* variant results in the best *speed* score and using only *critic* achieves the best *success rate 4–4*. Overall, adding *clue* has improved both *speed* and *success* rate while decreasing the *played rate*. Different from GPT models, *Claude* is able to play the game in some configurations while *Luminous* could not able to get produce any successful episodes.

TEMPLATE C.4.1

```
clue:display
guess_agreement:no
agreement_explanation:None of the letters in "shelf" match with the letters that could be present in the word based on the given guess feedback.
```

TEMPLATE C.4.2

```
guess:stand
explanation: A stand is an object used for displaying items. It has 5 letters.
```

Figure 14: Wordle prompt template for Player A (Guesser) to share critic's opinion in the Wordle Game Variant (+Clue, +Critic)

TEMPLATE C.4.3

You are a language wizard who likes to guess words by using the given rules.

Welcome to Wordle! You have six attempts to guess the target word, a valid English word of five lowercase letters (a-z). Please use the tags ”guess:” and ”explanation:” to provide a concise explanation for each guess.

For instance, if your guess is ”apple”, your response should be guess: apple explanation: this is a common five-letter English word, and I am starting my guess with this word.

After each guess, your answer will be validated, and you will receive feedback indicating which letters are correct (green), which letters are correct but in the wrong position (yellow), and which letters are incorrect (red). This feedback can be useful in determining which letters to include or exclude in your next guess.

For example, the feedback for ”apple” might be:
guess_feedback:
a ⟨yellow⟩ p ⟨yellow⟩ p ⟨green⟩ l ⟨yellow⟩ e ⟨red⟩

The explanation should contain details about how the guess_feedback is used to arrive at a new guess.

Let’s begin with your first guess.

TEMPLATE C.4.4

guess: hello
explanation: This is a common five-letter English word, and I am starting my guess with this word.

(a) Player A (Guesser) in the Wordle Game Basic Variant

TEMPLATE C.4.5

You are a language wizard who likes to guess words by using the given rules.

Welcome to Wordle! You have six attempts to guess the target word, a valid English word of five lowercase letters (a-z). Please use the tags ”guess:” and ”explanation:” to provide a concise explanation for each guess.

To help you make an informed guess, you will receive a clue for the word, such as clue: snowy white.

Here is an example guess based on the clue: guess: apple
explanation: In the fairy tail Snow White, the girl is killed because she eats a poisoned apple. And the word apple has 5 letters.

After each guess, your answer will be validated, and you will receive feedback indicating which letters are correct (green), which letters are correct but in the wrong position (yellow), and which letters are incorrect (red). This feedback can be useful in determining which letters to include or exclude in your next guess.

For example, the feedback for ”apple” might be:
guess_feedback:
a ⟨yellow⟩ p ⟨yellow⟩ p ⟨green⟩ l ⟨yellow⟩ e ⟨red⟩

The explanation should contain details about how both the guess_feedback and clue work together in tandem to formulate a new guess.

Let’s begin with your first guess.
clue:display

TEMPLATE C.4.6

guess: screen
explanation: A display can refer to a computer screen, and ”screen” is a valid English word with five letters.

(b) Player A (Guesser) in the Wordle Game Variant (+Clue)

Figure 15: Wordle prompt templates for basic and with clue variants

TEMPLATE C.4.7

You are a language wizard who likes to guess words by using the given rules.

Welcome to Wordle! You have six attempts to guess the target word, a valid English word of five lowercase letters (a-z). Please use the tags "guess:" and "explanation:" to provide a concise explanation for each guess.

To help you make an informed guess, you will receive a clue for the word, such as
clue: "snowy white"

Here is an example guess based on the clue:
guess: apple
explanation: In the fairy tail Snow White, the girl is killed because she eats a poisoned apple. And the word apple has 5 letters.

I will then indicate whether I agree or disagree with your guess and provide rationale, but agreeing with a guess does not confirm its correctness. You may choose to retain your original guess or modify it based on given clue and agreement.

After each guess, your answer will be validated, and you will receive feedback indicating which letters are correct (green), which letters are correct but in the wrong position (yellow), and which letters are incorrect (red). This feedback can be useful in determining which letters to include or exclude in your next guess.

For example, the feedback for "apple" might be:
guess_feedback:
a ⟨yellow⟩ p ⟨yellow⟩ p ⟨green⟩ l ⟨yellow⟩ e ⟨red⟩

Here are some rules to keep in mind: 1. Please do not generate any guess feedback or clue on your own 2. Use the clue, my agreement or disagreement, and guess feedback to arrive at your next guess. This should be reflected in your explanation as well.

Let’s begin with your first guess.
clue:display

TEMPLATE C.4.8

guess: shelf
explanation: A shelf is a common item used for displaying objects. It has 5 letters.

(a) Prompt template for Player A (Guesser) in the Wordle Game Variant (+Clue, +Critic)

TEMPLATE C.4.9

I need your assistance with a word game in which we need to find a 5-letter word using a clue, a guess and an explanation for the guess.

For example, if the clue is "snow white" and the guess is "apple", you will get the information like this:
clue:snow white
guess:apple
explanation:the fairy tale Snow White involves the girl eating a poisoned apple, which has 5 letters.

Your task is to either agree or disagree with my guess based on the given clue.

If you disagree, I will make another guess, and if you agree, I will proceed in the game using the same guess.

Initially, guess feedback is not available, but as the game proceeds, it becomes available. The guess feedback is not for current guess, but rather an indication of what letters can be present in the current guess. A letter highlighted in green indicates that it is a correct letter in the correct position, while yellow indicates that it is a correct letter in the wrong position, and red indicates a wrong letter. At this point, you should use all the provided information, including the clue, guess, explanation, and guess feedback, to decide whether the given guess matches the clue, and also stick to guess feedback or not.

Please respond in lowercase letters and stick to this format:
agreement:yes or no
explanation:your reason for doing so

Please note that you must not guess any word based on the riddle clue and stick to the given format while responding.

Let’s begin.

clue:display
guess:shelf
explanation:A shelf is a common item used for displaying objects. It has 5 letters.

TEMPLATE C.4.10

agreement: no
explanation: None of the letters in "shelf" match with the letters that could be present in the word based on the given guess feedback.

(b) Prompt template for Player B (Critic) in the Wordle Game Variant (+Clue, +Critic)

Figure 16: Wordle prompt templates for players with clue and critic variants

#	A	Game Master	B
1	[A⟩GM] guess: hello explanation: This is a common five-letter English word, and I am starting my guess with this word.		
2		[A⟨GM] guess_feedback: h<red> e<yellow> l<yellow> l<red> o<yellow>	
3	[A⟩GM] guess: world explanation: The feedback indicates that the letters "e" and "o" are correct but in the wrong position. The word "world" has these letters in the correct positions, so I am guessing this word.		
4		[A⟨GM] guess_feedback: w<red> o<green> r<red> l<yellow> d<yellow>	
5	[A⟩GM] guess:swirl explanation: The feedback indicates that the letters "o" is correct and in the correct position. The letter "l" is correct but in the wrong position. The word "swirl" has these letters in the correct positions, so I am guessing this word.		
6		[A⟨GM] <continues for remaining attempts>	
7		[GM|GM] attempts: 6 target_word = model guess: clerk guess_feedback: c<red> l<yellow> e<yellow> r<red> k<red>	
8		[GM|GM] game_result = LOSS	

(a) Traditional variant without clue and critic

#	A	Game Master	B
1	[A⟩GM] guess: rigid explanation: The clue "unbending" suggests a word that means inflexible or stiff. "Rigid" fits this description and has five letters.		
2		[A⟨GM] guess_feedback: r<red> i<yellow> g<red> i<red> d<red>	
3	[A⟩GM] guess:crisp explanation: The feedback shows that none of the letters in "rigid" are correct. The clue "unbending" could also describe something that is firm or fresh. "Crisp" fits this description and has five letters.		
4		[A⟨GM] guess_feedback: c<red> r<red> i<green> s<yellow> p<red>	
5	[A⟩GM] guess:spicy explanation: The feedback shows that the second letter is correct and in the right position. The clue "unbending" doesn't seem to be a direct clue for "spicy," but it could be interpreted as something that is strong or intense. "Spicy" fits this description and has five letters.		
6		[A⟨GM] <game flow continues>	
7		[GM|GM] attempts: 6 target_word = stiff target_word_clue = unbending guess: split guess_feedback: s<green> p<red> l<red> i<yellow> t<yellow>	
8		[GM|GM] game_result = LOSS	

(b) Variant with only clue

Figure 17: Excerpts of *wordle* game plays for the traditional variant (a) and the one with only clue (b) (GPT3.5/GPT3.5)

#	A	Game Master	B
1	[A⟩GM] guess: crook explanation: The word "uneven" makes me think of something that is not straight or has a bend, like a crook.		
2		[GM|GM] Share the guess information to critic	
3			[GM⟨B] agreement: no explanation: The guess "crook" has 5 letters, but the word "uneven" does not necessarily imply a bend or a curve. It could also mean something that is not equal or balanced.
4		[GM|GM] Critic disagrees with the Guesser – Sharing the critic's explanation with the guesser	
5	[A⟨GM] relays critic's opinion to the guesser		
6	[A⟩GM] guess: lumpy explanation: When I think of "uneven," I think of something that is not smooth or flat, like a lumpy surface. The word "lumpy" has 5 letters.		
7		[GM|GM] Change in player1's guess guess_before_critic_opinion: crook critic_agreement: no guess_after_critic_opinion: lumpy Proceeding with guess validation	
8	[A⟨GM] guess_feedback: l<red> u<red> m<red> p<red> y<green>		
9	[A⟩GM] guess:gravy explanation: The word "lumpy" makes me think of a food that has lumps in it, like gravy. The word "gravy" has 5 letters.		

(a)

#	A	Game Master	B
10			[GM⟩B] clue:uneven guess:gravy explanation:The word "lumpy" makes me think of a food that has lumps in it, like gravy. The word "gravy" has 5 letters. guess_feedback: l<red> u<red> m<red> p<red> y<green>
11			[GM⟨B] agreement: no explanation: The guess "gravy" does not match the clue "uneven". Additionally, the guess feedback does not contain any letters that match the letters in "uneven".
12		[GM|GM] Critic disagrees with the Guesser – Sharing the critic's explanation with the guesser	
13		[GM|GM] 'guess:': 'crinkl', 'explanation:': 'When I think of "uneven," I think of something that is not smooth or flat, like a crinkled surface. The word "crinkl" is not a valid English word, so I will try another guess.'	
14		[GM|GM] Guesser Error: INVALID_WORD_LENGTH while parsing Player 1's (model: gpt-3.5-turbo) response, retrying	
15		[GM|GM] Aborted due to invalid format in response	
16		[GM|GM] game_result = ABORTED	

(b)

Figure 18: Excerpt of *wordle* game play for the variant with clue and critic (divided into two parts due to spacing) (GPT3.5/GPT3.5)

D Drawing Game Details

In this game, the task is to draw a simple grid where Player A (Instruction Giver) needs to instruct Player B (Instruction Follower) how to draw, starting from an empty grid. The Game Master instructs the Player A to generate a referring expression that applies to the given *target grid*. The expression could include phrases that refer to the group of cells in a certain pattern or a single cell and includes additional information about the letter to be filled with. The Game Master passes the generated instruction to the Player B and instructs it to draw the grid that matches the given expression. In the first turn, the Player B starts initialises a grid with empty cells. An empty cell is indicated by the character "□", and a filled cell is an occurrence of any uppercase letter in the alphabet. The Player B applies the given expression to the current state of the grid and returns the result after each turn. The Player A continues to generate expressions until the filled cells in the target grid are described and the Player B keeps updating the current grid incrementally throughout the played turns in the game. The game finishes when Player A generates "DONE". As a fallback, the game also stops when the number of turns reaches the total number of cells in the target grid. The prompt templates for both players are given in Figure 19.

Instantiation We experiment with two different settings for *datasets* in this game called *compact* and *random* grids. Each dataset includes 20 different grids resulting in a total of 40 grids, which are 5x5. A **compact grid** stands for a grid with filled cells that follow a certain pattern. Ideally, such grids can be filled by describing the pattern in a single turn or less number of turns than by describing each filled cell one at a time. Each target grid includes at least five filled cells with the same letter (randomly selected for each instance). We manually defined 20 grids that have certain patterns, e.g. filled as M, cross, two rows are filled, three columns are filled, etc. A **random grid** is a randomly initialised grid where the cells do not follow a certain pattern when filled. Each target grid includes at least five and at most ten filled cells with the same letter (randomly selected for each instance). The location of each cell is randomly selected.

The main idea for having two different datasets is to test whether the evaluated language models can generate instructions that are compact (Player A side) and whether the generated instruction can be executed to obtain the drawing of the target grid (Player B side). Also, testing with random grids may reveal whether the game can be played with multiple turns by describing each filled cell one turn at a time.

Evaluation The evaluation of each episode is carried out by calculating three different measurement types.

1. **Target $\longleftrightarrow$ Drawn grid**: The comparison is done by comparing each filled cell in the target grid with the one at the same position in the drawn grid and calculate *Precision*, *Recall* and *F1-score*. At the turn level, we calculate these scores given the drawn grid up to that point. At the episode level, the drawn grid at the last turn is used. So the *incremental behaviour is to see an increase* in the scores after each interaction.

2. **Changed cell count**: We keep track of the number of cells that change after applying the given instruction on the Player B side. It reveals how certain generated expressions lead to the change of multiple cells, which can be an indication of *compact* instructions. At the turn level, it is simply the number of changed cells in the current state of the grid (after applying the instruction in the turn) with a comparison to the previous state of the grid. At the episode level, the number of changed cells at each turn is averaged.

3. **Generated instruction length**: it measures the number of characters in the generated instruction by the Player A at each turn. At the episode level, it is the average of number of characters in the generated instructions at each turn.

4. **Generated instruction token size**: it measures the average number of tokens in the generated instruction by the Player A at each turn. At the episode level, it is the average of number of characters in the generated instructions at each turn.

TEMPLATE D.0.1

```
Let us play a game. The goal is to fill an
empty grid that looks like this:
□ □ □ □ □
□ □ □ □ □
□ □ □ □ □
□ □ □ □ □
□ □ □ □ □

A filled grid below is 5 by 5 and can look like
this:
□ □ □ □ □
□ □ E □ □
□ □ □ □ □
□ □ □ □ □
X X X X X

I want you to describe this grid to me, step by
step. You don’t need to describe the empty
squares, which are denoted with ”□”. Only
describe the location of letters in the grid.
Then you wait for me to say "What is your next
instruction?", and then you continue with the
next step. Take the size of the grid into
consideration while giving instructions. When
you have described everything, you say "DONE".
For the filled grid above, here are the example
steps.

What is your next instruction?
Instruction: Put an E in second row third column

What is your next instruction?
Instruction: Fill the last row with X

What is your next instruction?
Instruction: DONE

Another example with the following 5 by 5 grid:
W □ □ □ □
□ W □ □ □
□ □ W □ □
□ □ □ W □
Z □ □ □ W

What is your next instruction?
Instruction: Put an W in five cells diagonally
starting from top left going to bottom right

What is your next instruction?
Instruction: Put Z in the last row first column

What is your next instruction?
Instruction: DONE

Ok. Please do this for the following example,
which is a 5 by 5 grid.

$TARGET_GRID_INSTANCE$
```

TEMPLATE D.0.2

```
What is your next instruction?
```

TEMPLATE D.0.3

```
Instruction: $INSTRUCTION$
```

(a) Template for Player A (Instruction Giver)

TEMPLATE D.0.4

```
Let us draw something together. There is an
empty grid with a size 5 by 5, like so:

□ □ □ □ □
□ □ □ □ □
□ □ □ □ □
□ □ □ □ □
□ □ □ □ □

I will give you instructions like "put
an X in the top left", and you return the
grid by applying the given instruction, like so:

Instruction: put an X in the top left

X □ □ □ □
□ □ □ □ □
□ □ □ □ □
□ □ □ □ □
□ □ □ □ □

Or for another instruction such as "fill the
fifth column with T", you return the updated
grid by applying the given instruction in all
places that the command corresponds to, like so:

Instruction: fill the fifth column with T

X □ □ □ T
□ □ □ □ T
□ □ □ □ T
□ □ □ □ T
□ □ □ □ T

Or for another instruction such as "fill
the fourth column second row with P", you
return the updated grid by applying the given
instruction in all places that the command
corresponds to, like so:

Instruction: fill the fourth column second row
with P

X □ □ □ T
□ □ □ P T
□ □ □ □ T
□ □ □ □ T
□ □ □ □ T

Now create an empty grid with a size 5 by 5 and
execute the following commands at each step.
Once you execute the command, return only the
grid and exclude all other text from the output.

Instruction: $INSTRUCTION$
```

TEMPLATE D.0.5

```
$GRID$
```

TEMPLATE D.0.6

```
Instruction: $INSTRUCTION$
```

(b) Template for Player B (Instruction Follower)

Figure 19: Drawing game prompt templates for players

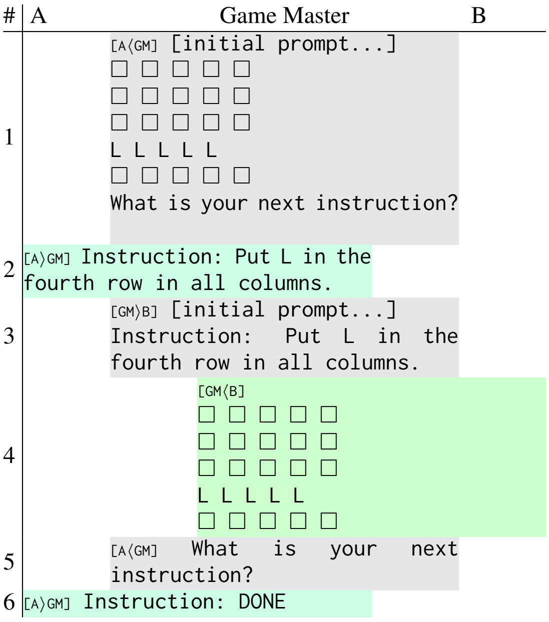
model	% played ↑	success rate ↑	F1 ↑
3–3	65	5	44
3.5—3.5	97	12.5	65
3.5-–4	97	15	70
4-–3.5	80	30	79
4–4	80	42.5	92
cl–cl	0	/	/
lm–lm	0	/	/

Table 5: A closer look at the models' performances for drawing: games played and the percentage of those ended successfully.

Results A closer look at the results (see Table 5) reveals that GPT4 is the best model in terms of reaching the highest *F1 score* the *success rate*. However, the *4–4* combination of players struggled in following the rules in comparison with *3.5–3.5*. *Luminous* and *Claude* models could not produce any output where they followed the rules.

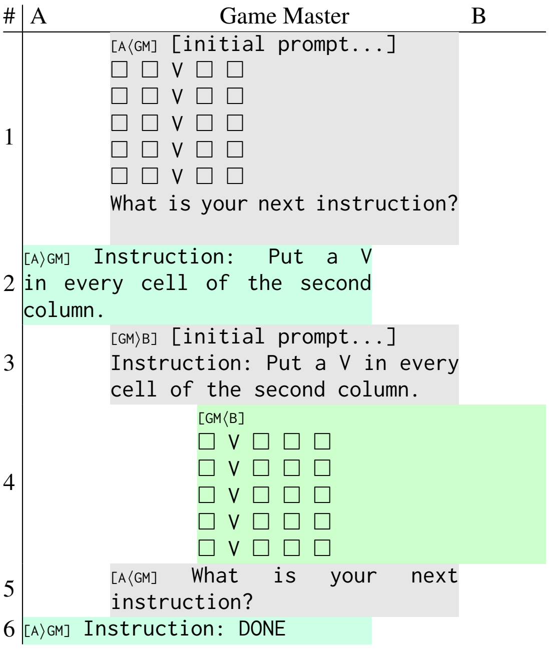
Example transcripts We present example transcripts for both compacts and random grids and the values of the evaluation metric.

- Example 1 (compact): Figure 20a
  - ○ Precision: 100
  - ○ Recall: 100
  - ○ F1: 100
  - ○ Changed cell count: 5
  - ○ Generated instruction length: 39
  - ○ Generated instruction token size: 9
  - ○ **Success: 1**
- Example 2 (compact): Figure 20b
  - ○ Precision: 0
  - ○ Recall: 0
  - ○ F1: 0
  - ○ Changed cell count: 5
  - ○ Generated instruction length: 43
  - ○ Generated instruction token size: 10
  - ○ **Success: 0**
- Example 3 (random): Figure 21a
  - ○ Precision: 100
  - ○ Recall: 100
  - ○ F1: 100
  - ○ Changed cell count: 1.3
  - ○ Generated instruction length: 44.6
  - ○ Generated instruction token size: 9.6
  - ○ **Success: 1**
- Example 4 (random): Figure 21b
  - ○ Precision: 50
  - ○ Recall: 50
  - ○ F1: 50
  - ○ Changed cell count: 1
  - ○ Generated instruction length: 35
  - ○ Generated instruction token size: 8
  - ○ **Success: 0**



#	A	Game Master	B
1		`[A⟨GM]` [initial prompt...] □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ L L L L L □ □ □ □ □ What is your next instruction?	
2	`[A⟩GM]` Instruction: Put L in the fourth row in all columns.		
3		`[GM⟩B]` [initial prompt...] Instruction: Put L in the fourth row in all columns.	
4			`[GM⟨B]` □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ L L L L L □ □ □ □ □
5		`[A⟨GM]` What is your next instruction?	
6	`[A⟩GM]` Instruction: DONE		

(a) Successful gameplay from both players



#	A	Game Master	B
1		`[A⟨GM]` [initial prompt...] □ □ V □ □ □ □ V □ □ □ □ V □ □ □ □ V □ □ □ □ V □ □ What is your next instruction?	
2	`[A⟩GM]` Instruction: Put a V in every cell of the second column.		
3		`[GM⟩B]` [initial prompt...] Instruction: Put a V in every cell of the second column.	
4			`[GM⟨B]` □ V □ □ □ □ V □ □ □ □ V □ □ □ □ V □ □ □ □ V □ □ □
5		`[A⟨GM]` What is your next instruction?	
6	`[A⟩GM]` Instruction: DONE		

(b) Player A fails in generating the correct instruction (points to second column instead of third as in the target grid)

Figure 20: Excerpts of *drawing* game plays for *compact* examples (GPT3.5/GPT3.5)

#	A	Game Master	B
1		[A⟨GM] [initial prompt...] ☐ F ☐ ☐ ☐ ☐ ☐ ☐ F F ☐ ☐ ☐ ☐ ☐ F ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ What is your next instruction?	
2	[A⟩GM] Instruction: Put an F in the first row second column.		
3		[GM⟩B] [initial prompt...] Instruction: Put an F in the first row second column.	
4			[GM⟨B] ☐ F ☐
5		[A⟨GM] What is your next instruction?	
6	[A⟩GM] Instruction: Put two Fs in the second row fourth and fifth columns.		
7		[GM⟩B] Instruction: Put two Fs in the second row fourth and fifth columns.	
8			[GM⟨B] ☐ F ☐ ☐ ☐ ☐ ☐ ☐ F F ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐
9		[A⟨GM] What is your next instruction?	
10	[A⟩GM] Instruction: Put an F in the fourth row first column.		
11		[GM⟩B] Instruction: Put an F in the fourth row first column.	
12			[GM⟨B] ☐ F ☐ ☐ ☐ ☐ ☐ ☐ F F ☐ ☐ ☐ ☐ ☐ F ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐
13		[A⟨GM] What is your next instruction?	
14	[A⟩GM] Instruction: DONE.		

(a) Successful gameplay from both players

#	A	Game Master	B
1		[A⟨GM] [initial prompt ...] ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ C ☐ ☐ C ☐ ☐ ☐ ☐ ☐ C ☐ ☐ ☐ ☐ C ☐ What is your next instruction?	
2	[A⟩GM] Instruction: Put a C in second row fifth column.		
3		[GM⟩B] [initial prompt ...] Instruction: Put a C in second row fifth column.	
4			[GM⟨B] ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ C ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐
5		[A⟨GM] What is your next instruction?	
6	[A⟩GM] Instruction: Put a C in third row third column.		
7		[GM⟩B] Instruction: Put a C in third row third column.	
8			[GM⟨B] ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ C ☐ ☐ C ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐
9		[A⟨GM] What is your next instruction?	
10	[A⟩GM] Instruction: Put a C in fourth row second column.		
11		[GM⟩B] Instruction: Put a C in fourth row second column.	
12			[GM⟨B] ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ C ☐ ☐ C ☐ ☐ ☐ ☐ C ☐ ☐ ☐ ☐ ☐ ☐ ☐
13		[A⟨GM] What is your next instruction?	
14	[A⟩GM] Instruction: Put a C in fifth row second column.		
15		[GM⟩B] Instruction: Put a C in fifth row second column.	
16			[GM⟨B] ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐ C ☐ ☐ C ☐ ☐ ☐ ☐ C ☐ ☐ ☐ ☐ C ☐ ☐
17		[A⟨GM] What is your next instruction?	
18	[A⟩GM] Instruction: DONE		

(b) Player A & B fail in generating the correct instructions and drawing the matching grid

Figure 21: Excerpts of *drawing* game plays for *random* examples (GPT3.5/GPT3.5)

E Character Picture Reference Game Details

The Game Master selects a target and two distractor grids and instructs the Player A to generate a referring expression that uniquely describes the target grid and differentiates it from the distractors. There is a history of work on referring expression generation and the topic has recently received new attention in the context of neural learners. The Game Master then provides the same three grids and the referring expression from Player A to Player B. The three grids are numbered such as *first*, *second*, and *third* and the order of grids are randomly shuffled for Player B. Player B generates a single expression that should refer to the number of the target grid that matches the given expression. The game is played for a single turn. The prompt templates for both players are given in Figure 22.

Instantiation We manually created target grids and apply a number of edits on them to obtain two distractors. A single edit is essentially choosing a random filled cell and converting it into an empty cell. We apply the following two configurations to create the dataset with 36 instances for experimenting with this game.

1. **Edit distance of two**: We apply one or two edits to the target grid to obtain a distractor grid. We created 18 such tuples of a target and two distractor grids using two edits.
2. **Edit distance of four**: We apply the same idea explained above but create 18 grids with four edits.

We want to to measure whether the tested language models are able to differentiate between grids that look a like (two edit distances) and whether it is simpler compared to grids that somewhat look slightly different (four edit distances).

Evaluation The evaluation of each episode is done by checking whether the Player B guesses the target grid correctly. It is simply “successful” when the generated expression matches the number of the target grid and “failed” otherwise. Additionally, we also measure the number of characters and the token size in the referring expression generated by the Player A.

Results A closer look at the results for the game (see Table 6 reveals that almost all models (except *Luminous*) can follow the rules where GPT *3.5* and *4* models reach a perfect score for the played games ratio. For the success ratio, *Claude* achieves the highest score followed by *4*.

model	% played ↑	success ↑
3–3	83	63
3.5—3.5	100	66
3.5-–4	100	72
4-–3.5	100	61
4–4	100	77
cl–cl	100	88
lm–lm	0	/

Table 6: A closer look at the models’ performances for drawing: games played and the percentage of those ended successfully.

Example transcripts We present example transcripts for both compacts and random grids and the values of the evaluation metric.

- Example 1 (edit distance 2): Figure 23a
  - Generated expression length: 52
  - Generated expression token size: 9
  - **Success: 1**
- Example 2 (edit distance 2): Figure 23b
  - Generated expression length: 23
  - Generated expression token size: 4
  - **Success: 0**
- Example 3 (edit distance 4): Figure 24a
  - Generated expression length: 29
  - Generated expression token size: 7
  - **Success: 1**
- Example 4 (edit distance 4): Figure 24b
  - Generated expression length: 32
  - Generated expression token size: 7
  - **Success: 0**

TEMPLATE E.0.1

```
Let us play a game. You are given three grids
where each of them is 5 by 5 in size. Grids
have empty cells marked with "□" and filled
cells marked with "X". The goal is to generate
a single referring expression that captures the
main content in the grid named as "Target grid".
Generate the referring expression starting with
the tag "Expression: " for the given target
grid and exclude any other text.

Here is an example with grids. The first grid
is the target grid and the following two grids
are distractors.

Target grid:

X X X X X
□ □ X □ □
□ □ X □ □
□ □ X □ □
□ □ X □ □

Second grid:

X X X X X
□ □ X □ □
X X X X X
□ □ X □ □
X X X X X

Third grid:

X X X X X
□ □ □ □ □
□ □ □ □ □
□ □ □ □ □
X X X X X

The referring expression for the given target
grid is like so:
Expression: Filled as T.

Ok. Now do this for the following grids.

Target grid:
$TARGET_GRID$

Second grid:
$SECOND_GRID$

Third grid:
$THIRD_GRID$

Generate the referring expression for the given
target.
```

TEMPLATE E.0.2

```
Expression: $EXPRESSION$
```

(a) Prompt template for Player A (Instruction Giver) in the Reference Game.

TEMPLATE E.0.3

```
Let us play a game. You are given three grids
where each of them is 5 by 5 in size. Grids
have empty cells marked with "□" and filled
cells marked with "X". You are also given a
referring expression that describes one of the
given grids. The goal is to select a grid that
matches the given referring expression. Here
is an example with grids and the referring
expression. Generate only the number (in text)
of the grid that the given expression matches
to by selecting first, second, or third. Start
with the tag "Answer: " and followed by the
generated expression.

First grid:
X X X X X
□ □ X □ □
X X X X X
□ □ X □ □
X X X X X

Second grid:
X X X X X
□ □ X □ □
□ □ X □ □
□ □ X □ □
□ □ X □ □

Third grid:
X X X X X
□ □ □ □ □
□ □ □ □ □
□ □ □ □ □
X X X X X

Expression: Filled as T.
Question: Which grid does the expression refer
to? Generate only the names of the grids like
"first", "second" or "third", exclude any other
word.
Answer: second

Ok. Now do this for the following grids.
Generate only the number (in text) of the
grid that the given expression matches to by
selecting first, second, or third.

First grid:
$FIRST_GRID$

Second grid:
$SECOND_GRID$

Third grid:
$THIRD_GRID$

Expression: $EXPRESSION$
Question: Which grid does the expression refer
to? Generate only the names of the grids like
"first", "second" or "third", exclude any other
word.
```

TEMPLATE E.0.4

```
Answer: $ANSWER$
```

(b) Prompt template for Player B (Instruction Follower) in the Reference Game

Figure 22: Reference game prompt templates for players

#	A	Game Master	B
1		[A⟨GM] [initial prompt...] Target grid: X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X Second grid: X ☐ X ☐ X ☐ X ☐ X ☐ ☐ ☐ X ☐ X ☐ X ☐ ☐ ☐ X ☐ X ☐ X Third grid: X ☐ X ☐ X ☐ X ☐ X ☐ ☐ ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ ☐ Generate the referring expression for the given target.	
2	[A⟩GM] Expression: Alternating X and empty cells in a diagonal pattern.		
3		[GM⟩B] [initial prompt...] First grid: X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X Second grid: X ☐ X ☐ X ☐ X ☐ X ☐ ☐ ☐ X ☐ X ☐ X ☐ X ☐ X ☐ X ☐ ☐ Third grid: X ☐ X ☐ X ☐ X ☐ X ☐ ☐ ☐ X ☐ X ☐ X ☐ ☐ ☐ X ☐ X ☐ X Expression: Alternating X and empty cells in a diagonal pattern. Question: Which grid does the expression refer to? Generate only the names of the grids like "first", "second" or "third", exclude any other word.	
4			[GM⟨B] Answer: first

(a) Successful gameplay from both players

#	A	Game Master	B
1		[A⟨GM] [initial prompt ...] Target grid: X X X X X X ☐ ☐ ☐ X X ☐ X ☐ X X ☐ ☐ ☐ X X X X X X Second grid: X X X X X X ☐ ☐ ☐ X X ☐ ☐ ☐ X X ☐ ☐ ☐ X X ☐ X X X Third grid: X ☐ X ☐ X X ☐ ☐ ☐ X X ☐ X ☐ X X ☐ ☐ ☐ X X X X X X Generate the referring expression for the given target.	
2	[A⟩GM] Expression: Filled as X in a cross shape.		
3		[GM⟩B] [initial prompt ...] First grid: X ☐ X ☐ X X ☐ ☐ ☐ X X ☐ X ☐ X X ☐ ☐ ☐ X X X X X X Second grid: X X X X X X ☐ ☐ ☐ X X ☐ ☐ ☐ X X ☐ ☐ ☐ X X ☐ X X X Third grid: X X X X X X ☐ ☐ ☐ X X ☐ X ☐ X X ☐ ☐ ☐ X X X X X X Expression: Filled as X in a cross shape. Question: Which grid does the expression refer to? Generate only the names of the grids like "first", "second" or "third", exclude any other word.	
4			[GM⟨B] Answer: third

(b) Player A & B fail in generating the correct expression and guessing the correct grid

Figure 23: Excerpts of *reference* game plays for *edit distance 2* examples (GPT3.5/GPT3.5)

#	A	Game Master	B
1		[A⟨GM] [initial prompt...] Target grid: X X X X X X □ □ □ X X □ X □ X X □ □ □ X X X X X X Second grid: X X X X X X □ □ □ □ X □ □ □ X X □ □ □ □ X X □ X X Third grid: □ X □ X □ X □ □ □ X X □ X □ X X □ □ □ □ X X X X X Generate the referring expression for the given target.	
2	[A⟩GM] Expression: Filled as X in a cross shape.		
3		[GM⟩B] [initial prompt...] First grid: X X X X X X □ □ □ □ X □ □ □ X X □ □ □ □ X X □ X X Second grid: □ X □ X □ X □ □ □ X X □ X □ X X □ □ □ □ X X X X X Third grid: X X X X X X □ □ □ X X □ X □ X X □ □ □ X X X X X X Expression: Filled as X in a cross shape. Question: Which grid does the expression refer to? Generate only the names of the grids like "first", "second" or "third", exclude any other word.	
4			[GM⟨B] Answer: third

(a) Successful gameplay from both players

#	A	Game Master	B
1		[A⟨GM] [initial prompt...] Target grid: □ □ X □ □ □ □ X □ □ X X X X X □ X □ X □ □ □ X □ □ Second grid: □ □ X □ □ □ □ □ □ □ X X □ X □ □ X □ □ □ □ □ X □ □ Third grid: □ □ □ □ □ □ □ X □ □ X □ □ □ X □ X □ X □ □ □ X □ □ Generate the referring expression for the given target.	
2	[A⟩GM] Expression: Filled as a cross in the center.		
3		[GM⟩B] [initial prompt...] First grid: □ □ X □ □ □ □ X □ □ X X X X X □ X □ X □ □ □ X □ □ Second grid: □ □ □ □ □ □ □ X □ □ X □ □ □ X □ X □ X □ □ □ X □ □ Third grid: □ □ X □ □ □ □ □ □ □ X X □ X □ □ X □ □ □ □ □ X □ □ Expression: Filled as a cross in the center. Question: Which grid does the expression refer to? Generate only the names of the grids like "first", "second" or "third", exclude any other word.	
4			[GM⟨B] Answer: second

(b) Player A & B fail in generating the correct expression and guessing the correct grid

Figure 24: Excerpts of *reference* game plays for *edit distance 4* examples (GPT3.5/GPT3.5)

F Scorekeeping Game Details

In an interaction, one device of the *conversational grounding* anchoring process is that participants coordinate what is private knowledge and what information has already been shared in previous turns. After each utterance, the status of novel information should be updated from private to shared in both agents' discourse models. This is how they do *scorekeeping*, *i.e.* keeping track of the *common ground* which is built incrementally, turn by turn (Clark and Brennan, 1991; Lewis, 1979).

For example, consider a slot-filling conversation with asymmetric roles between a questioner and an answerer, which can occur as part of customer service, job interviews or medical diagnosis interactions. If the questioner asks *Where do you work?*, at this point this is typically private information that only the answerer knows. After the reply, the place of work becomes shared information, and both the questioner and the answerer know that.

The evaluation method for scorekeeping proposed by Madureira and Schlangen (2022) is to probe, after each turn, whether the dialogue model's representations correctly encode information about the private or shared status of true and false statements. With cLLMs, we can instead probe by directly posing side questions to an agent while it interacts with another agent.

We thus introduce a dialogue game which enables testing the scorekeeping abilities of these models, by measuring how well the cLLM's discourse model gets correctly updated after each turn.

Game Description This is a slot-filling conversation, mediated by a game master, with asymmetric roles between a questioner and an answerer. We define n slots to be filled. The answerer player A privately knows the values of all slots from the beginning of the interaction (passed via an initial prompt) but the questioner Q does not. The questioner then asks n questions, one by one, aiming at filling those slots based on A's answers. A final state is reached when Q fills all the slots and the the goal state is having all values correctly filled. Before the interaction starts and after each question-answer pair, the game master probes the agent's discourse model by asking about the status (private or shared) of every slot, one by one and in a random order, in the conversation so far. This results in a sequence of $n + 1$ probing rounds, each containing n binary decisions, which can be used to evaluate the performance of the model.

Instantiation Here we introduce two versions of this setting with 5 slots: i) a travel agent and a customer booking a trip and ii) a recruiter and a job applicant in a job interview. We implement the questioner programmatically and let the cLLM play the role of the answerer. This game is an example of a "messenger" setup, where the game master plays a more active role, by parsing responses and performing the probing rounds. The game master begins by instructing the cLLM about the setting, explaining that it should give replies according to the given values for each slot, as shown in Templates 25 and 26.

Besides the task-oriented requests from Q, the cLLM must also respond to probing questions privately posed by the game master. The initial prompt defines special labels to be used for each type of question and response. Because the questioner's order of requests is under the control of the game master, the truth values are known and can be immediately compared to the cLLM' answers. For completeness, we also make the probing before any move from the questioner. Note that, in the first probing round, all slot values are private, whereas in the last one, all are shared.

(i) Travel Agency: simulates a conversation between a customer (the cLLM) and a travel agent. The customer wishes to book a trip according to a set of 5 slots: `from` (origin), `to` (destination), `by` (means of transportation), `class` and `when` (time of departure). An example of the initial prompt and the instance is shown in Template 25. For probing, the game master can ask, for instance, *"Does the travel agent know where you want to go?"*. The correct answer is *no* until the travel agent has received a reply for that slot, when the correct answer changes to *yes*.

(i) Job Interview: simulates a conversation between a job applicant (the cLLM) and a recruiter. The job applicant has a CV with 5 slots: `bachelor`, `industry experience`, `highest education`, `other skills` and `availability`. An example of the initial prompt and the instance is shown in Template 26. For probing, the game master can ask, for instance, *"Has the recruiter been informed about your availability?"*. Again, the correct answer is *no* until the recruiter has received a reply for that slot.

Implementation For each version, we generate 10 instances by randomly selecting values for all slots and a random order for the questioner's requests. The cLLM is prompted to only give short, direct answers to avoid that slot values are given in anticipation. In slot filling turns, if the agent uses the wrong tag, the game is aborted immediately. We consider that a slot was filled if the answer contains its value. We also check whether it contains any new value and update the probing ground truth accordingly. In probing rounds, the game master prompts the model to answer yes or no. If, for some reason, it was not possible to parse a valid response during probing, we add additional instructions for clarity in the request. After the maximum number of 5 failed attempts, an invalid response symbol is used instead and the game will be aborted after that probing round is finished. Each probing question is posed on its own and does not get appended to the dialogue context in subsequent turns. For instance, after (q_i, a_i), the $i+1$-th sequence of probes is made. At request $i+1$, however, the dialogue context contains only the game questions and answers up to turn i and none of the probes.

Evaluation Besides following the game instructions, a competent player should i) provide the correct slot value to answer each question accordingly; and ii) know, at any point, which slot values have already been disclosed to the questioner and what has not yet been revealed. Specifically, the exact turn when a slot value shifts from private to shared should be correctly detected.

Turn-Level Scores: At each turn, the game master collects n binary answers (yes or no). We thus use **accuracy** as a turn-level score, computed by comparing these n answers to the corresponding n truth values. An ideal model would achieve high accuracy at all turns. We also track a binary label which is 1 if the current slot is correctly given in the answer.

Episode-Level Scores: At the end of an episode, $(n+1)n$ answers have been collected via probing. We compute **accuracy** across all answers. However, given that this is a binary classification task, the random performance is very high. We thus also compute **Cohen's** κ (Cohen, 1960) as an episode-level score, truncated at 0.

As discussed in (Madureira and Schlangen, 2022), a model biased towards considering all values as private would perform well at initial turns, whereas models biased towards shared would perform well at final turns. We follow their suggestion to also evaluate the performance in middle turns, where the distribution of labels is more balanced. For that, we report the accuracy at the third probing round, namely **middle-accuracy**.

The validity of the results rely on the slots having been correctly filled. As a sanity check, we compute the proportion of answers that contain the correct slot value as an additional episode level score (**slot-filling-accuracy**).[10]

Preferred Score: The harmonic mean between slot-filling-accuracy and truncated κ is normalised to $[0, 100$ and used as the main score, summarising the performance of an agent in an episode.

Example Interactions Figures 27 and 28 shows the main interaction and one round of probing for the Claude model, with metadata about whether the answers were correct.

---

[10] However, even if the cLLM hallucinates an answer, the probing can still be performed, because a wrong value is still a shared value.

TEMPLATE F.0.1

```
WHAT: Travel
FROM: London
TO: Stuttgart
BY: Train
CLASS: Economy
WHEN: In May
```

TEMPLATE F.0.2

```
You are a customer of a travel agency. Here is
a description of the details of the travel you
want to make:

$INSTANCE$

The travel agent does not know about it yet.

Questions from the travel agent will start with
TRAVEL-AGENT. Please reply in the form: ANSWER:
<some text>

I will also ask you questions. These questions
will start with ME: .  Please answer these
questions with: ASIDE: <some text>

Important: Give short, direct answers!

Let us start.
```

TEMPLATE F.0.3

```
TRAVEL-AGENT: $AGENT-QUESTION%
```

TEMPLATE F.0.4

```
ANSWER: $ANSWER$
```

TEMPLATE F.0.5

```
ME: %CG-QUESTION%
```

TEMPLATE F.0.6

```
ASIDE: %CG-REPLY%
```

Figure 25: Travel agency version of the PRIVATE-SHARED dialogue game. From top: Example Instance, Initial Prompt for Customer, Next-Round Template for Main Task, Response Parsing Schema for Customer Action, Next-Round Template for Probing Task, Response Parsing Schema for Reply to Probing Question.

TEMPLATE F.0.7

```
WHAT: Job Application
BACHELOR: Music
INDUSTRY-EXPERIENCE: two years
HIGHEST-EDUCATION: Bachelor
OTHER-SKILLS: French
AVAILABILITY: Last week of January
```

TEMPLATE F.0.8

```
You are an applicant in a job interview.  Here
is a description of your CV:

$INSTANCE$

The recruiter does not know about it yet.
Questions from the recruiter will start with
RECRUITER. Please reply in the form:  ANSWER:
<some text>

I will also ask you questions. These questions
will start with ME:  .  Please answer these
questions with: ASIDE: <some text>

Important: Give short, direct answers!

Let us start.
```

TEMPLATE F.0.9

```
RECRUITER: $AGENT-QUESTION%
```

TEMPLATE F.0.10

```
ANSWER: $ANSWER$
```

TEMPLATE F.0.11

```
ME: %CG-QUESTION%
```

TEMPLATE F.0.12

```
ASIDE: %CG-REPLY%
```

Figure 26: Job Interview version of the PRIVATE-SHARED dialogue game. From top: Example Instance, Initial Prompt for Job Applicant, Next-Round Template for Main Task, Response Parsing Schema for Customer Action, Next-Round Template for Probing Task, Response Parsing Schema for Reply to Probing Question.

#	A	Game Master	B
1		[A⟨GM] [initial prompt]	
2		[GM⟩B] What is the next request?	
3			[GM⟨B] Where are you going to?
4		[A⟨GM] TRAVEL-AGENT: Where are you going to?	
5	[A⟩GM] ANSWER: Dresden		
6		[GM|GM] Dresden	
7		[GM|GM] Slot filled: True	
8		[GM⟩B] Dresden	
9		[GM⟩B] What is the next request?	
10			[GM⟨B] What is the origin of your trip?
11		[A⟨GM] TRAVEL-AGENT: What is the origin of your trip?	
12	[A⟩GM] ANSWER: Copenhagen		
13		[GM|GM] Copenhagen	
14		[GM|GM] Slot filled: True	
15		[GM⟩B] Copenhagen	
16		[GM⟩B] What is the next request?	
17			[GM⟨B] What kind of transportation do you want to use?
18		[A⟨GM] TRAVEL-AGENT: What kind of transportation do you want to use?	
19	[A⟩GM] ANSWER: Plane		
20		[GM|GM] Plane	
21		[GM|GM] Slot filled: True	
22		[GM⟩B] Plane	
23		[GM⟩B] What is the next request?	
24			[GM⟨B] What is the date of departure?
25		[A⟨GM] TRAVEL-AGENT: What is the date of departure?	
26	[A⟩GM] ANSWER: First week of June		
27		[GM|GM] First week of June	
28		[GM|GM] Slot filled: True	
29		[GM⟩B] First week of June	
30		[GM⟩B] What is the next request?	
31			[GM⟨B] Do you have a preference for class?
32		[A⟨GM] TRAVEL-AGENT: Do you have a preference for class?	
33	[A⟩GM] ANSWER: First or business		
34		[GM|GM] First or business	
35		[GM|GM] Slot filled: True	
36		[GM⟩B] First or business	

Figure 27: Excerpt of the scorekeeping game for Claude.

#	A	Game Master	B
1		[GM|GM] Begin probing	
2		[A⟨GM] ME: Does the travel agent already know your origin? Please answer yes or no.	
3	[A⟩GM] ASIDE: No		
4		[GM|GM] no	
5		[GM|GM] Answer for FROM valid after 1 tries.	
6		[GM|GM] Answer is incorrect.	
7		[A⟨GM] ME: Has the travel agent been informed about your destination? Please answer yes or no.	
8	[A⟩GM] ASIDE: Yes		
9		[GM|GM] yes	
10		[GM|GM] Answer for TO valid after 1 tries.	
11		[GM|GM] Answer is correct.	
12		[A⟨GM] ME: Has the travel agent been informed about your preferred means of transportation? Please answer yes or no.	
13	[A⟩GM] ASIDE: No		
14		[GM|GM] no	
15		[GM|GM] Answer for BY valid after 1 tries.	
16		[GM|GM] Answer is correct.	
17		[A⟨GM] ME: Have you informed the travel agent about your class preference? Please answer yes or no.	
18	[A⟩GM] ASIDE: No		
19		[GM|GM] no	
20		[GM|GM] Answer for CLASS valid after 1 tries.	
21		[GM|GM] Answer is correct.	
22		[A⟨GM] ME: Is the travel agent aware of the dates of your trip? Please answer yes or no.	
23	[A⟩GM] ASIDE: No		
24		[GM|GM] no	
25		[GM|GM] Answer for WHEN valid after 1 tries.	
26		[GM|GM] Answer is correct.	
27		[GM|GM] End probing	

Figure 28: Probing at round 3 for Claude.