A Llama Sunk My Battleship! Asking Rational Questions with LLMs via Bayesian Inference

Gabriel Grand[1,2*] **Valerio Pepe**[3] **Joshua B. Tenenbaum**[1,2] **Jacob Andreas**[1]
[1]MIT CSAIL [2]MIT BCS [3]Harvard SEAS

Abstract

One of the hallmarks of an intelligent agent is the ability to ask good questions. While facility with language is clearly a prerequisite, even in simple settings, LLMs can struggle to come up with questions that yield useful information—suggesting a failure of grounded reasoning. We study this phenomenon in a question-asking task based on the classic board game *Battleship*, where both text-only and multimodal LLMs perform far below human baselines. We propose a Bayesian model that combines a LLM-driven prior over questions with a probabilistic world model to facilitate coherent reasoning. We find that with a surprisingly modest sample budget for "mental computation," our method is well-calibrated to human performance across varied *Battleship* board scenarios. Notably, this approach allows much smaller LLMs, such as CodeLlama-7b, to perform on par with GPT-4. These results support the emerging trend toward test-time inference as a scaling route for LLM reasoning, while highlighting the utility of probabilistic world models for grounding and structuring such computations.

1 Introduction

Considerable efforts are being directed towards optimizing large language models (LLMs) to answer human queries, with the aim of building helpful and aligned AI assistants (Ouyang et al., 2022; Bai et al., 2022; Rafailov et al., 2024). However, a key aspect of human reasoning is our ability to *ask* questions in order to reduce our uncertainty about the world and inform our future actions (Graesser et al., 1993; Markant and Gureckis, 2012; Hawkins et al., 2015). LLMs are clearly linguistically capable of expressing an unbounded space of questions. However, even in simple grounded environments, coming up with *informative* questions typically requires evaluating possible world states and their relative probabilities—and it is less clear whether current LLMs are well-suited to this kind of System 2-like reasoning.

This paper explores what it would take to build models capable of asking informative questions at a human-like level in an environment that is minimalistic but combinatorially complex. We adopt a cognitively-inspired approach that models people as *resource rational agents* (Anderson, 1990; Chater and Oaksford, 1999; Lieder and Griffiths, 2019) subject to strict computational constraints. Behavioral studies reveal that both children and adults are "greedy" information-seekers in active learning and consider only a few hypotheses at a time (Klayman and Ha, 1989; Vul et al., 2014; Markant et al., 2016; Meder et al., 2019; Ruggeri et al., 2016; Cheyette et al., 2023). AI assistants face similar pressures in production settings, where there are practical limits to how much time a model can take to "think" of a response. Accordingly, we focus on the computational tradeoff between *informativity* and *resource efficiency* in a setting where the goal is to obtain the most information with as few questions as possible.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

*Correspondence to gg@mit.edu. Code and data available at github.com/gabegrand/battleship.

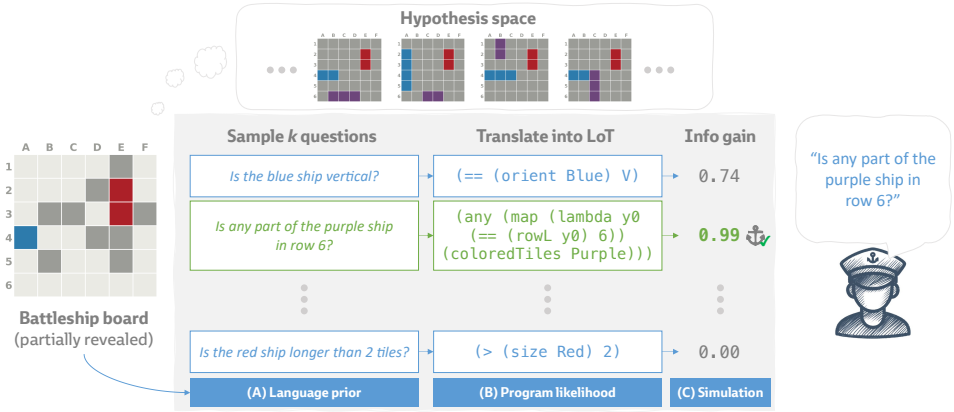


Figure 1: How much "mental computation" is needed to ask rational questions in grounded environments, like the board game *Battleship*? We introduce a Bayesian model that integrates LLMs with a probabilistic world model to perform sample-based inference. Given a partially-revealed board, our model (A) samples k questions from a LLM and (B) translates these into programs in a simple DSL. (C) The utility of a question is computed by simulating the program against a hypothesis space of boards consistent with the observation. Here, the best question achieves Expected Information Gain (EIG) of 0.99, meaning the answer would rule out nearly half the boards in the hypothesis space. This approach successfully avoids asking LLM-generated questions that are uninformative (e.g., "Is the red ship longer than 2 tiles?") and efficiently attains human performance with a modest sampling budget.

We explore several computational approaches in an adaptation of the classic board game *Battleship* where, in addition to firing at tiles, players can ask open-ended questions to seek information about the board. In our experiments, we compare two different LLMs (CodeLlama-7b and GPT-4/GPT-4V) as well as a classical grammar baseline. While these models by themselves yield questions that are only weakly informative relative to human-authored questions, we find that they *can* be effective as components in a Bayesian question-asking model with the ability to perform sample-based inference in an internal world model. We introduce an overarching framework called Langugage-Informed Program Sampling (Fig. 1) where LLMs play two distinct roles: (A) as priors over questions, and (B) as language-conditional distributions that map questions into executable code expressions. By varying the number of questions we sample from the prior, we can control how much "mental computation" LIPS performs. We find that for surprisingly small values of k, LIPS yields informative questions that are well-calibrated to human data. These results illustrate that factorizing AI agents into separate language-generation and world-modeling modules can lead to efficient, resource-rational solutions to general classes of reasoning problems.

2 The Battleship Game

We adopt the *Battleship* task developed by Rothe et al. (2017, 2018), a grid-based environment that evaluates participants' ability to ask goal-directed questions. In this task, participants were presented a partially-revealed board (Fig. 1) and asked to come up with a **natural language question** that would help to reveal the locations of the hidden ships, with the constraint that the question should admit a single-word answer. The task consists of 18 unique 6x6 board contexts, each containing three ships (red, blue, and purple) of varying length (2-4 tiles), orientation (horizontal or vertical), and placement. While later variants extended the paradigm to study multi-turn interactions (Rothe et al., 2019), here we focus on the original, single-turn task and discuss ongoing extensions to the multi-turn setting in §5.

3 Models

Following prior work, we begin by considering an ideal observer model of a player that starts with a uniform prior $p(s)$ over possible boards consistent with the observed initial state. After asking a

question x and receiving an answer y, the player performs a Bayesian update to their belief distribution

$$p(s \mid y; x) = \frac{p(y \mid s; x)p(s)}{\sum_{s' \in S} p(y \mid s'; x)p(s')} \tag{1}$$

where the likelihood $p(y \mid s; x)$ is 1 if y is consistent with s and 0 otherwise. The marginal likelihood can be computed by enumeration or approximated by sampling over a hypothesis space of boards S.

The player's uncertainty about the hidden state of the game board can be measured by the Shannon entropy $H(s)$ (Shannon, 1948), and the value of a question x can be defined as its Expected Information Gain (EIG):

$$\text{EIG}(x) = H(s) - \sum_{y \in Y_x} p(y \mid x) H(s \mid x, y) \tag{2}$$

Intuitively, EIG provides a log-space measure of the number of candidate boards that the player can rule out with question x. For instance, an ideal yes/no question that rules out 50% of possible boards would achieve $\text{EIG}(x) = 1$. (Throughout, we use $\log_2$, so EIG is measured in bits.)

In *Battleship*, questions that admit a large set of possible answers, denoted Y_x, can achieve $\text{EIG}(x) \gg 1$ (e.g., "What is the top-left corner of the red ship?"). However, some answers may be more informative than others; this uncertainty gives rise to the expectation over possible answers in Eq. 2.

In the work by (Rothe et al., 2017), EIG was considered as one among several heuristic features (complexity, answer type, etc.) in a Boltzmann energy model that was fit to maximize the likeihood of the collected human questions. Here we take a complementary approach: instead of fitting our model to human data collected from *Battleship*, we instead aim to sample directly from a distribution of maximally-informative questions—*without* positing the space of features these questions might have. We hypothesize that human-like questions will fall out naturally from a Bayesian model with a very generic prior that is subject to cognitive resource constraints.

3.1 Language-Informed Program Sampling (LIPS)

We formulate our model as a simple Monte Carlo search with a parameter k that controls the amount of internal computation the model performs. (We are in part inspired by the bounded space model of Ullman et al., 2016 for creative language generation.) Given some proposal distribution over questions, we sample k questions and choose the one that maximizes EIG:

$$\{x_1, \ldots, x_k\} \sim p(x \mid s) \tag{3}$$

$$x^* = \arg\max_{x_i} \text{EIG}(x_i) \tag{4}$$

A central challenge of this approach is choosing a suitable proposal distribution $p(x \mid s)$ that admits efficient sampling. Moreover, as the notation implies, this distribution should ideally be *board-conditional* so as to generate targeted questions about the particular board at hand. To facilitate computation of EIG, it is also critical to have a proposal distribution that is capable of expressing questions as code expressions that can be deterministically executed against the board following some denotational semantics; i.e., $y = [\![x]\!]_s$. We consider two kinds of question-proposal distribution that allow us to instantiate our model.

3.2 Grammar proposal distribution

As a "classical" baseline, we consider a probabilistic context-free grammar (Johnson, 1998) as a proposal distribution over questions. We adopt the grammar of Rothe et al. (2017)[2] whose rules and terminals correspond to key concepts in *Battleship*: ships vary in *color*, *size*, *orientation*, *location*, etc. The grammar also encodes numeric and set-theoretic operations to support comparisons; e.g., "How many of the blue ship's tiles are in column B?"

[2]See Table SI-1 in Rothe et al. (2017) for the full grammar. We omit λ-abstractions, which rarely yield well-formed questions during sampling, and we filter out trivial expressions of depth 1.

Answer → Bool | Num. | Color | Orient. | Loc.
Bool → 'T' | 'F' | (and B B) | (touch Ship Ship) . . .
Num. → 0 | 1 | . . . | 9 | (+ N N) | . . .
Num. → (size Ship) | (row L) | (col L) . . .
Color → Ship |
Ship → 'Blue' | 'Red' | 'Purple'
Orient. → 'Horizontal' | 'Vertical' | (orient Ship)
Loc. → $1A$ | $1B$ | . . . | $6F$ | (topleft Set) . . .
Set → (tiles Color) | (∩ Set Set) | 'AllColors' . . .

Figure 2: PCFG for grammar proposal distribution baseline.

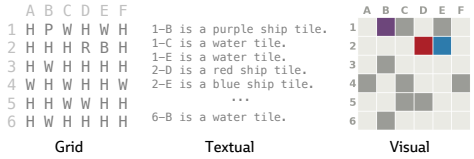


Figure 3: We experiment with 3 different board representations: an ASCII-style grid, a textual serialization, and a visual image encoded in a multimodal prompt to GPT-4(V).

3.3 Language model proposal distribution

A recent line of work in probabilistic programming explores using Large Language Models (LLMs) as instantiations of humanlike priors in Bayesian models (Lew et al., 2020, 2023; Dohan et al., 2022; Ellis, 2023). LLMs represent an attractive question proposal distribution for several reasons. First, since they are trained on vast corpora of natural text, LLMs directly encode a prior over plausible questions. Moreover, LLMs are strong in-context learners (Brown et al., 2020) and are increasingly amenable to instruction from the experimenter (Ouyang et al., 2022; Rafailov et al., 2024). Consequently, by constructing an appropriate prompt (detailed in §4), we can transform a generic LLM into a proposal distribution over questions in the *Battleship* domain. This approach faces two main challenges, which we discuss below.

Grounding generation in the state of the world Ideally, we would like our model to be "stimulus computable" (Yamins and DiCarlo, 2016), accepting the same images and task instructions as a human participant. While multimodal LLMs are growing in popularity and availability (Driess et al., 2023; OpenAI, 2023b), it remains unclear to what extent they are capable of extracting structured visual information—such as a *Battleship* board—into an appropriate computational representation. We experiment with three different types of board representation (Fig. 3) in order to evaluate the degree to which our LLM proposal distributions are able to leverage board-conditional information.

Translating from natural language to code LIPS posits that the question-asker mentally draws and evaluates k samples and chooses the most informative one. This is straightforward in the case of the PCFG, which directly generates programs, but not for the case of LLMs, which output natural language. To address this, we follow the approach of the *Rational Meaning Construction* framework (Wong and Grand et al., 2023), which uses LLMs to implement a "meaning function" that translates from natural language into code. Concretely, we decompose the LLM proposal into separate **question generation** $p(l \mid s)$ and **language-to-program translation** $p(x \mid l)$ distributions, which we approximate via sampling.

$$p(x \mid s) = \sum_{l} p(x \mid l) p(l \mid s) \tag{5}$$

This formalization admits many possible denotational semantics—$[\![\cdot]\!]_s$ could be implemented by a LISP interpreter, a Python program, or even a LLM. For convenience, we use the same *Battleship* DSL from Rothe et al. (2017), which allows us to take advantage of the fast C++ implementation of the EIG function developed for that work.[3]

4 Experiment

4.1 Participants, materials, and methods

Human data We use the human dataset collected by Rothe et al. 2017, which consists of 26-39 questions for each board composed by a single pool of N=40 participants, for a total of 605 question-board pairs. Participants were not "prompted" with any example questions; they were only given

[3] https://github.com/anselmrothe/EIG

Model	EIG		% Valid		% Informative		Program Depth		Program Size		Question Words	
	μ	σ_M	μ	σ_M	μ	σ_M	μ	σ_M	μ	σ_M	μ	σ_M
Human	1.27	0.04	1.00	0.00	0.97	0.01	3.22	0.07	4.51	0.14	7.12	0.08
Grammar	0.36	0.00	1.00	0.00	0.38	0.00	3.01	0.00	5.13	0.01	–	–
CodeLlama-7b	0.65	0.02	0.75	0.01	0.45	0.01	2.64	0.02	3.24	0.04	6.66	0.04
GPT-4 (few-shot)	0.77	0.02	0.88	0.01	0.59	0.01	2.61	0.02	3.22	0.04	6.23	0.03
GPT-4 (zero-shot)	0.66	0.01	0.40	0.01	0.35	0.01	3.73	0.04	5.04	0.09	5.19	0.02
GPT-4 (no board)	0.60	0.02	0.68	0.01	0.43	0.01	3.08	0.03	4.12	0.07	6.28	0.03

Table 1: Summary statistics of the underlying samples ($k = 1$) across all board contexts. Questions that translated to a parseable program are considered Valid, and those that achieved EIG > 0 are considered Informative. Program Depth and Size refer to the depth and number of nodes of the program abstract syntax tree. Question Words measures the number of words in the natural language question. μ and σ_M denote sample mean and standard error, respectively.

the constraint that the question should admit a single-word answer. As the program annotations in this dataset used an earlier version of the DSL, we manually translated a representative subset of the questions into the latest DSL and used a LLM to annotate the remaining programs.

LLMs We queried GPT-4 (OpenAI, 2023a,b) via API, using `gpt-4-0613` for the textual and grid board formats, and `gpt-4-vision-preview` for the visual format. To compare against a reproducible, open-source LLM, we used CodeLlama (Roziere et al., 2023), a member of the Llama 2 family of models that was finetuned for code generation. We obtained the model weights from HuggingFace (`CodeLlama-7b-hf`) and used the smallest variant of the model, which contains 7B parameters. We performed local inference on a single GPU, taking advantage of the `hfppl` library (Lew et al., 2023) to speed up inference via caching.

Prompting We fed both LLMs identical sets of algorithmically-constructed prompts (see the "Prompts" section in the Appendix). For question generation $p(l \mid s)$, each prompt consisted of instructions describing the task setup ("You are playing the board game Battleship. There are three ships on the board..."). In the **zero-shot** condition, the prompt concluded with a target game board (Fig. 3) and text to elicit a question. In the **few-shot** condition, the prompt additionally included 3 example boards, each with 10 questions from the human data. The example boards and questions were sampled without replacement in a leave-one-out manner so as to exclude human data collected for the target board. For translation $p(x \mid l)$, the prompt consisted of a similar task instruction, followed by 12 (l, x) pairs randomly sampled from the human data in the same manner.

Sampling For each LLM condition, we sampled 100 questions/board $\times$ 18 boards. To explore the effects of prompt and board formats, we repeated this process for each combination of {zero-shot, few-shot} $\times$ {textual, grid, visual, no board} using GPT-4(V). For the PCFG, which is not board-conditional, we sampled a single set of 100K questions and computed their EIG values for each board. Following Ullman et al. (2016), to avoid expensive re-collection of data, samples were grouped *post-hoc* into buckets of size k. Since the underlying samples are i.i.d., this provides an unbiased estimate of the true sampler, with the caveat that the effective sample size diminishes with k. Throughout, null hypothesis testing was conducted between conditions using Welch's t-test.

4.2 Results and Discussion

Informativity How informative are the questions collected from humans? And to what extent do our models capture the information-seeking quality of human questions? We computed EIG values for all human and model-generated questions (Table 1). Across the 18 boards, the average human question scored EIG $= 1.27$, while the best human question achieved considerably higher EIG $= 3.61$. Despite this large range, *virtually all* (97%) of the human questions were informative (EIG > 0), revealing that participants were highly sensitive to the board state.

In contrast, the underlying proposal distributions ($k = 1$) were substantially noisier than people: questions from CodeLlama and GPT-4 averaged EIG $= 0.65$-0.66, respectively, while questions from the grammar averaged EIG $= 0.36$. However, as Fig. 4 (top left) reveals, LIPS allows for a significant boost in performance: with just $k = 5$ samples, both LLMs approached human mean performance; and at $k = 10$, both models significantly outperformed the human mean, with $p < 0.001$ for CodeLlama, and $p = 0.01$ for GPT-4 (textual, few-shot). This trend continues for sample sizes $k = 20$ and $k = 50$, though all models still fall short of the best human-generated questions.

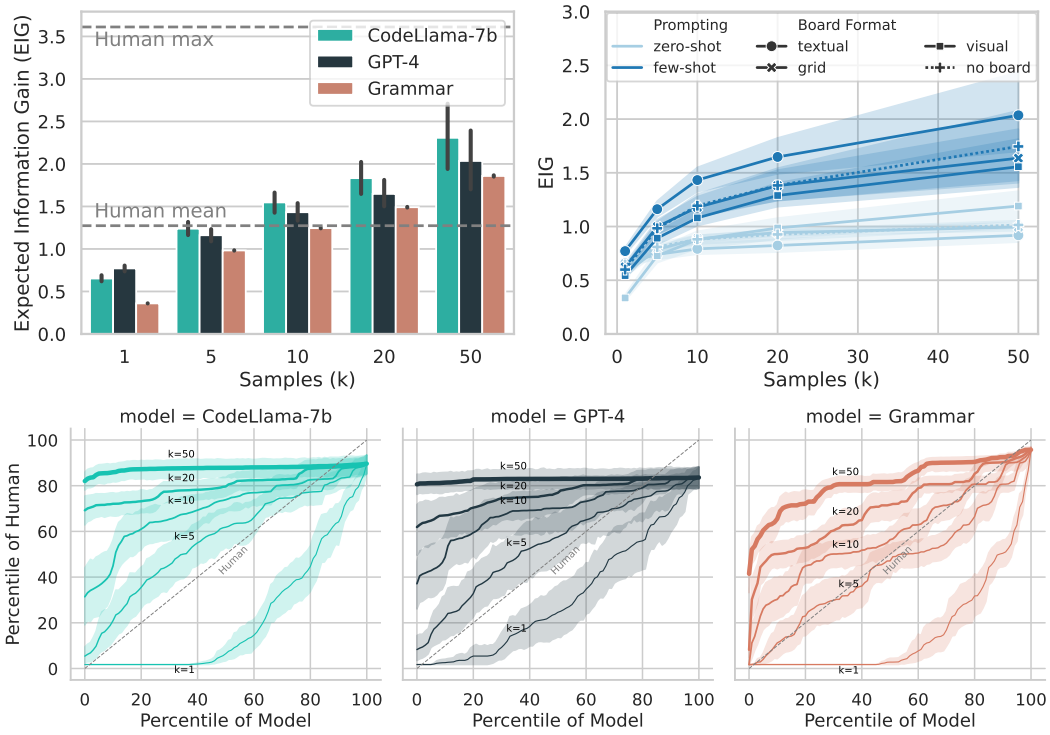


Figure 4: Comparing the informativity of model-generated questions against human data. **(Top left)** LIPS with two LLMs and a hand-engineered grammar as proposal distributions over questions. As k increases, all three models reach mean-human performance, though they fall short of the best human-generated questions. **(Top right)** Evaluating GPT-4's performance with different prompt formats and board representations. Including few-shot examples universally boosts EIG. However, performance varies depending on the board format. Notably, GPT-4(V) was unable to utilize the board's structure in text (grid) or images (visual), implying a failure of grounding. **(Bottom)** Q-Q plots comparing model vs. human EIG values at varying sample sizes. At $k = 5$, all three models are generally well-calibrated to humans, though they fall short of the top 10-20% of human questions. Throughout, error bars and shaded regions indicate 95% bootstrapped confidence intervals. GPT-4 and CodeLlama-7b refer to the few-shot, textual condition unless otherwise noted.

Sample efficiency What represents a cognitively-plausible amount of mental sampling? Fig. 4 (bottom) compares the full distribution of model vs. human EIG values for varying values of k. At $k = 5$, both LLMs were closely calibrated to the human distribution, performing on par with the grammar, which was hand-engineered to capture this distribution. In other words, the Nth percentile of human question-askers wrote questions that were of comparable informativity to the Nth percentile of samples from the model. However, the top human questions (approx. 85-90th percentile) outperformed the top model-generated questions.

Translation fidelity One restriction of our evaluation is that, in order for a question to be considered informative, it needs to be expressable in the *Battleship* DSL. But how effective is the model at translating questions into programs? As Table 1 (% Valid) shows, a high percentage of samples from CodeLlama (75%) and GPT-4 (88%) were successfully translated. Only in the GPT-4 (zero-shot) case did the translation model achieve low fidelity (40%). Since the model does not receive any examples in the zero-shot case, it is not surprising that many of the questions from this distribution were not translatable.

Groundedness To what extent did the LLM-generated questions take the board state into account? Of the valid programs sampled from each model, 40% (CodeLlama) and 33% (GPT-4) were *uninformative* (EIG $= 0$). This occurs when a question is redundant with respect to information already revealed in the board. (For instance, "Is the red ship vertical?" is uninformative for 3/18 boards in the stimulus set.) The high proportion of uninformative programs highlights a potential failure of

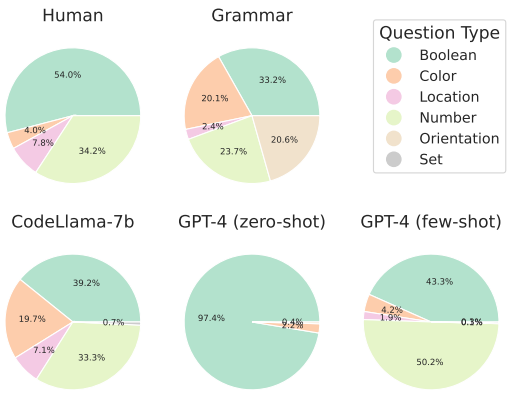


Figure 5: Proportion of top-level question types generated by each proposal distribution at $k = 1$.

grounding. Our evaluation of different board formats, shown in Fig. 4 (top right), provides further evidence of this issue. Of the four board formats (Fig. 3), the "textual" representation was the only one that significantly outperformed the "no board" condition ($p < 0.05$ for $k =$ 1-20). Notably, across k, the "visual" board format performed either significantly worse ($p < 0.05$ for $k = 1, 5$) or was not significantly different than the "no board" condition ($p > 0.05$ for $k =$ 10-50). These results show that that GPT-4V was unable to utilize the board's structure to formulate informative questions relative to a board-agnostic baseline.

Question type What *kinds* of information do humans ask about, and do the models reflect this distribution? As illustrated in Fig. 5, humans ask a diverse range of question types, with a preference for boolean and numeric answers. Owing to its structure, the grammar generates an approximately uniform distribution over types. Meanwhile, both of the few-shot prompted LLMs approximate the human distribution, though CodeLlama mirrors it more closely than GPT-4. Without access to examples, GPT-4 (zero-shot) defaults to boolean questions that echo traditional *Battleship* moves; e.g., "Is there a ship at 2-C?" Thus, the different choices of prior encode different inductive biases—and LLMs provide an especially flexible way of encoding both human general knowledge and domain-specific priors into Bayesian models.

5 Ongoing Work

While the *Battleship* task that we consider here provides a controlled setting for studying question-asking in humans and AI models, this initial toy problem has several key limitations. In human-AI interactions in the wild, information-seeking takes place iteratively over multiple dialogue turns. Furthermore, there is an explore/exploit trade-off between question asking and acting; AI assistants must consider the opportunity costs of asking for clarifying information from a user. Finally, unlike in a single-turn setting where certain questions are generally useful across contexts, over the course of a user-agent interaction, the *kinds* of questions that are useful will necessarily evolve as more information about the world state becomes available.

As a next step towards exploring some of these dynamics, we have developed an extension of our task environment to a synchronous two-player, multi-turn information seeking game. In this setting, which we call *Collaborative Battleship* (Fig. 6), players alternate between the roles of **Captain** and **Spotter**, working together to reveal the hidden ships in as few moves as possible. On each turn, the Captain must decide whether take action, by firing at a tile, or seek information, by asking a question to the Spotter, who has full visibility of the board but can only respond with a Yes/No answer.

We are currently piloting this experiment with pairs of human players on Prolific to understand how people navigate the trade-offs between asking questions and taking actions. In parallel, we are developing AI models to play both the Captain and the Spotter roles leveraging the modeling concepts detailed above: namely, the ability to perform Bayesian inference in a mental world model via conditional sampling. We are excited about the opportunities this extended setting affords for exploring rational LLM-powered models that can collaborate with human users to ask and answer questions in a grounded environment.

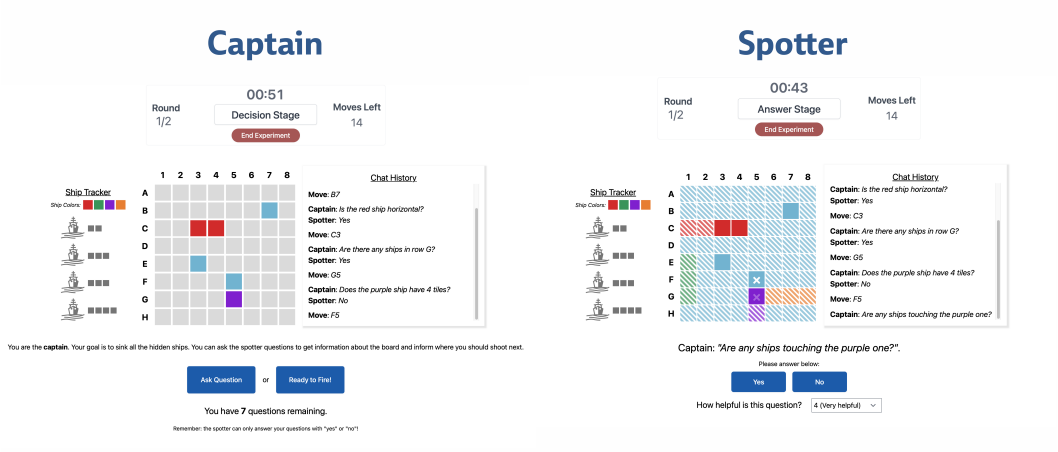


Figure 6: User interface for our *Collaborative Battleship* experiment, which evaluates information-seeking in a two-player synchronous dialogue environment. The Captain must choose between taking actions and asking questions given limited visibility, while the Spotter sees the whole board, but can only respond with Yes/No. In order to play at human-level, an AI agent must incorporate grounded language understanding (to answer questions), reasoning about uncertainty (to ask informative questions), and strategic decision making (to balance explore/exploit trade-offs).

6 Conclusion

As more and more people interact with language models on a daily basis, asking questions to efficiently clarify uncertainty will become an increasingly important capability for such systems. In this work, we introduced a new approach to building this kind of question-asking behavior by sampling questions from a noisy LLM prior and translating into code expressions. But where does this programming language come from in the first place? Here, we used an existing DSL as initial step, but our approach could be combined with Bayesian program induction techniques to learn a new DSL from data (Ellis et al., 2021; Wong et al., 2021; Grand et al., 2024; Piantadosi et al., 2024). Relaxing our assumptions even further, we might eschew a DSL in favor of a domain-general programming language like Python (Ellis, 2023; Wang et al., 2024).

Given the general abilities of LLMs on many reasoning tasks, it is also natural to ask in what settings intermediate symbolic representations—i.e., programs—are needed for effective information-seeking. A growing line of concurrent work on clarifying user preferences in a human-LLM dialogue context provides an interesting space of alternatives. Proposed approaches include structured prompting strategies both without (Li et al., 2023) and with Bayesian priors (Handa et al., 2024); self-improvement via finetuning on reasoning chains (Zelikman et al., 2022; Andukuri et al., 2024); and search over future dialogue turns (Piriyakulkij et al., 2023; Zhang and Choi, 2023). This latter view aligns most closely with our approach, which supports the emerging trend towards adaptive inference-time computation with LLMs. In this paper, we take the idea one step further by studying information-seeking in the context of a minimal but combinatorially-complex environment, demonstrating the efficiency advantages of grounded reasoning in a symbolic world model. We believe that this approach, when paired with LLMs' ability to *construct* such world models on-the-fly via code generation, offers a promising and powerful approach to scaling AI reasoning to novel problem domains.

7 Acknowledgments

We thank Brenden Lake, Robert Hawkins, Judy Fan, Noah Goodman, Guy Davidson, Sam Cheyette, Belinda Li, Maddy Bowers, and Lionel Wong for helpful discussions and feedback on this work.

The authors gratefully acknowledge support from the MIT Quest for Intelligence, the MIT-IBM Watson AI Lab, the Intel Corporation, AFOSR, DARPA, and ONR. GG is supported by the National Science Foundation (NSF) under Grant No. 2141064. JA is supported by NSF Grant IIS-2144855. JBT received support from AFOSR Grant #FA9550-19-1-0269, the MIT-IBM Watson AI Lab, ONR Science of AI and the DARPA Machine Common Sense program. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of sponsors.

References

Anderson, J. R. (1990). *The adaptive character of thought*. Psychology Press.

Andukuri, C., Fränken, J.-P., Gerstenberg, T., and Goodman, N. D. (2024). Star-gate: Teaching language models to ask clarifying questions. *arXiv preprint arXiv:2403.19154*.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., et al. (2022). Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language Models are Few-Shot Learners. *arXiv:2005.14165 [cs]*. arXiv: 2005.14165.

Chater, N. and Oaksford, M. (1999). Ten years of the rational analysis of cognition. *Trends in cognitive sciences*, 3(2):57–65.

Cheyette, S. J., Callaway, F., Bramley, N. R., Nelson, J. D., and Tenenbaum, J. (2023). People seek easily interpretable information. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 45.

Dohan, D., Xu, W., Lewkowycz, A., Austin, J., Bieber, D., Lopes, R. G., Wu, Y., Michalewski, H., Saurous, R. A., Sohl-Dickstein, J., et al. (2022). Language model cascades. *arXiv preprint arXiv:2207.10342*.

Driess, D., Xia, F., Sajjadi, M. S., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., Yu, T., et al. (2023). Palm-e: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*.

Ellis, K. (2023). Human-like few-shot learning via bayesian reasoning over natural language. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Ellis, K., Wong, C., Nye, M., Sablé-Meyer, M., Morales, L., Hewitt, L., Cary, L., Solar-Lezama, A., and Tenenbaum, J. B. (2021). Dreamcoder: Bootstrapping inductive program synthesis with wake-sleep library learning. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pages 835–850.

Graesser, A. C., Langston, M. C., and Baggett, W. B. (1993). Exploring information about concepts by asking questions. In *Psychology of Learning and Motivation*, volume 29, pages 411–436. Elsevier.

Grand, G., Wong, L., Bowers, M., Olausson, T. X., Liu, M., Tenenbaum, J. B., and Andreas, J. (2024). LILO: Learning interpretable libraries by compressing and documenting code. In *The Twelfth International Conference on Learning Representations*.

Handa, K., Gal, Y., Pavlick, E., Goodman, N., Andreas, J., Tamkin, A., and Li, B. Z. (2024). Bayesian preference elicitation with language models. *arXiv preprint arXiv:2403.05534*.

Hawkins, R. X., Stuhlmüller, A., Degen, J., and Goodman, N. D. (2015). Why do you ask? good questions provoke informative answers. In *CogSci*.

Johnson, M. (1998). PCFG models of linguistic tree representations. *Computational Linguistics*, 24(4):613–632.

Klayman, J. and Ha, Y.-w. (1989). Hypothesis testing in rule discovery: Strategy, structure, and content. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 15(4):596.

Lew, A. K., Tessler, M. H., Mansinghka, V. K., and Tenenbaum, J. B. (2020). Leveraging unstructured statistical knowledge in a probabilistic language of thought. In *Proceedings of the Annual Conference of the Cognitive Science Society*.

Lew, A. K., Zhi-Xuan, T., Grand, G., and Mansinghka, V. K. (2023). Sequential monte carlo steering of large language models using probabilistic programs. *arXiv preprint arXiv:2306.03081*.

Li, B. Z., Tamkin, A., Goodman, N., and Andreas, J. (2023). Eliciting human preferences with language models. *arXiv preprint arXiv:2310.11589*.

Lieder, F. and Griffiths, T. L. (2019). Resource-rational analysis: Understanding human cognition as the optimal use of limited computational resources. *Behavioral and Brain Sciences*, 43.

Markant, D. and Gureckis, T. (2012). Does the utility of information influence sampling behavior? In *Proceedings of the annual meeting of the cognitive science society*, volume 34.

Markant, D. B., Settles, B., and Gureckis, T. M. (2016). Self-directed learning favors local, rather than global, uncertainty. *Cognitive science*, 40(1):100–120.

Meder, B., Nelson, J. D., Jones, M., and Ruggeri, A. (2019). Stepwise versus globally optimal search in children and adults. *Cognition*, 191:103965.

OpenAI (2023a). GPT-4 Technical Report.

OpenAI (2023b). GPT-4V(ision) System Card.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Piantadosi, S. T., Rule, J. S., and Tenenbaum, J. B. (2024). Learning as Bayesian inference over programs. In Griffiths, T. L., Chater, N., and Tenenbaum, J. B., editors, *Bayesian Models of Cognition: Reverse-engineering the Mind*. MIT Press.

Piriyakulkij, T., Kuleshov, V., and Ellis, K. (2023). Active preference inference using language models and probabilistic reasoning. *arXiv preprint arXiv:2312.12009*.

Rafailov, R., Sharma, A., Mitchell, E., Manning, C. D., Ermon, S., and Finn, C. (2024). Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36.

Rothe, A., Lake, B. M., and Gureckis, T. (2017). Question asking as program generation. *Advances in neural information processing systems*, 30.

Rothe, A., Lake, B. M., and Gureckis, T. M. (2018). Do people ask good questions? *Computational Brain & Behavior*, 1:69–89.

Rothe, A., Lake, B. M., and Gureckis, T. M. (2019). Asking goal-oriented questions and learning from answers. In *CogSci*, pages 981–986.

Roziere, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Remez, T., Rapin, J., et al. (2023). Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*.

Ruggeri, A., Lombrozo, T., Griffiths, T. L., and Xu, F. (2016). Sources of developmental change in the efficiency of information search. *Developmental psychology*, 52(12):2159.

Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

Ullman, T. D., Siegel, M. H., Tenenbaum, J., and Gershman, S. (2016). Coalescing the vapors of human experience into a viable and meaningful comprehension. In *CogSci*.

Vul, E., Goodman, N., Griffiths, T. L., and Tenenbaum, J. B. (2014). One and done? optimal decisions from very few samples. *Cognitive science*, 38(4):599–637.

Wang, R., Zelikman, E., Poesia, G., Pu, Y., Haber, N., and Goodman, N. (2024). Hypothesis search: Inductive reasoning with language models. In *The Twelfth International Conference on Learning Representations*.

Wong, C., Ellis, K., Tenenbaum, J. B., and Andreas, J. (2021). Leveraging language to learn program abstractions and search heuristics. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 11193–11204. PMLR.

Wong, L., Grand, G., Lew, A. K., Goodman, N. D., Mansinghka, V. K., Andreas, J., and Tenenbaum, J. B. (2023). From word models to world models: Translating from natural language to the probabilistic language of thought. *arXiv preprint arXiv:2306.12672*.

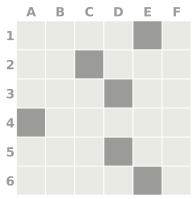
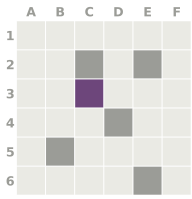
Yamins, D. L. and DiCarlo, J. J. (2016). Using goal-driven deep learning models to understand sensory cortex. *Nature neuroscience*, 19(3):356–365.

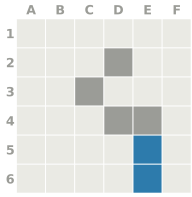
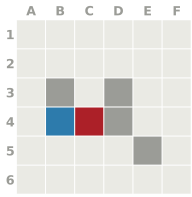
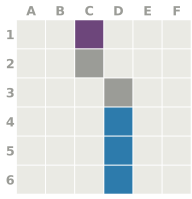
Zelikman, E., Wu, Y., Mu, J., and Goodman, N. (2022). Star: Bootstrapping reasoning with reasoning. *Advances in Neural Information Processing Systems*, 35:15476–15488.

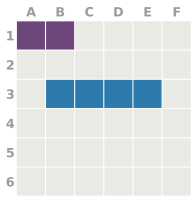
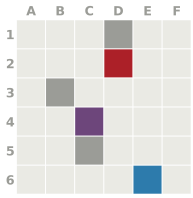
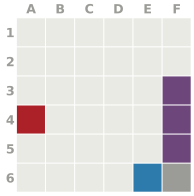
Zhang, M. J. and Choi, E. (2023). Clarify when necessary: Resolving ambiguity through interaction with lms. *arXiv preprint arXiv:2311.09469*.

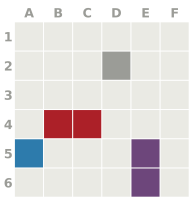
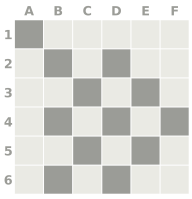
8 Appendix

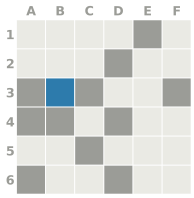
Table 2: Examples questions and programs for each board context. For each model, we sampled one best ⋆ and one random ⚅ question. For humans, the random sample was drawn from the full pool of participant data for each trial; for models, the random sample was selected from the LIPS outputs with $k = 10$, which provides a close match to mean human performance. In cases where the best EIG value was attained by multiple programs, tiebreaking was random. Note that the Grammar generates programs directly—many of which are not readily translatable to natural language—so "Question" is omitted for this model.

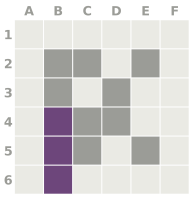
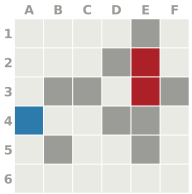
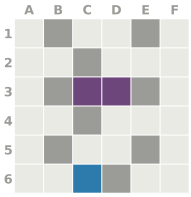
Board	Model		Question	Program	EIG
Trial 1 	Human	⋆	At what location is the top left part of the red ship?	`(topleft (coloredTiles Red))`	4.67
		⚅	Is there a purple tile at 1A?	`(== (color 1A) Purple)`	0.39
	CodeLlama	⋆	At what location is the top left part of the blue ship?	`(topleft (coloredTiles Blue))`	4.67
		⚅	How many tiles is the blue ship?	`(size Blue)`	1.36
	GPT-4	⋆	What is the location of one blue tile?	`(topleft (coloredTiles Blue))`	4.67
		⚅	How many tiles is the blue ship?	`(size Blue)`	1.36
	Grammar	⋆	—	`(topleft (coloredTiles Red))`	4.67
		⚅	—	`(color 6B)`	1.40
Trial 2 	Human	⋆	What is the location of one red tile?	`(topleft (coloredTiles Red))`	4.58
		⚅	How many tiles is the purple ship?	`(size Purple)`	1.58
	CodeLlama	⋆	What is the location of one red tile?	`(topleft (coloredTiles Red))`	4.58
		⚅	How many tiles is the red ship?	`(size Red)`	1.41
	GPT-4	⋆	Where is one blue tile located?	`(topleft (coloredTiles Blue))`	4.58
		⚅	How many tiles is the purple ship?	`(size Purple)`	1.58
	Grammar	⋆	—	`(bottomright (union (intersection (set AllTiles) (coloredTiles Red)) (intersection (unique (set...`	4.65
		⚅	—	`(color 3F)`	1.43

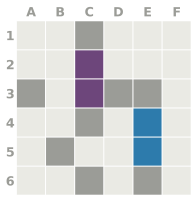
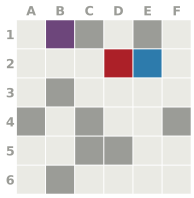
Board	Model		Question	Program	EIG
Trial 3 	Human	⋆	At what location is the top left part of the purple ship?	`(topleft (coloredTiles Purple))`	4.62
		⚅	How many tiles is the red ship?	`(size Red)`	1.44
	CodeLlama	⋆	How many tiles is the purple ship?	`(size Purple)`	1.44
		⚅	Is the red ship horizontal?	`(== (orient Red) H)`	0.99
	GPT-4	⋆	How many tiles is the red ship?	`(size Red)`	1.44
		⚅	Is the red ship horizontal?	`(== (orient Red) H)`	0.99
	Grammar	⋆	—	`(bottomright (unique (intersection (set AllTiles) (coloredTiles Purple))))`	4.73
		⚅	—	`(orient Purple)`	0.99
Trial 4 	Human	⋆	At what location is the top left part of the purple ship?	`(topleft (coloredTiles Purple))`	4.62
		⚅	At what location is the top left part of the purple ship?	`(topleft (coloredTiles Purple))`	4.62
	CodeLlama	⋆	How many tiles is the red ship?	`(size Red)`	1.57
		⚅	How many tiles is the red ship?	`(size Red)`	1.57
	GPT-4	⋆	How many tiles is the red ship?	`(size Red)`	1.57
		⚅	How many tiles is the red ship?	`(size Red)`	1.57
	Grammar	⋆	—	`(bottomright (intersection (set AllTiles) (intersection (coloredTiles Purple) (set AllTiles))))`	4.64
		⚅	—	`(== (orient Blue) H)`	0.91
Trial 5 	Human	⋆	At what location is the top left part of the red ship?	`(topleft (coloredTiles Red))`	4.66
		⚅	How many tiles is the purple ship?	`(size Purple)`	1.57
	CodeLlama	⋆	At what location is the bottom right part of the purple ship?	`(bottomright (coloredTiles Purple))`	1.90
		⚅	How many tiles is the purple ship?	`(size Purple)`	1.57
	GPT-4	⋆	How many tiles is the purple ship?	`(size Purple)`	1.57

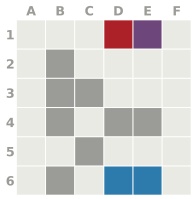
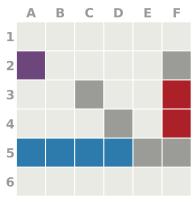
Board	Model		Question	Program	EIG
		⚅	How many tiles is the purple ship?	`(size Purple)`	1.57
	Grammar	⋆	—	`(topleft (unique (coloredTiles Red)))`	4.66
		⚅	—	`(size Purple)`	1.57
Trial 6 	Human	⋆	At what location is the top left part of the red ship?	`(topleft (coloredTiles Red))`	4.73
		⚅	Does the red ship touch both other ships?	`(and (touch Red Blue) (touch Red Purple))`	0.60
	CodeLlama	⋆	At what location is the top left part of the red ship?	`(topleft (coloredTiles Red))`	4.73
		⚅	Is the red ship 2 tiles long?	`(== (size Red) 2)`	1.00
	GPT-4	⋆	Where is a tile of the red ship?	`(topleft (coloredTiles Red))`	4.73
		⚅	How many tiles is the red ship?	`(size Red)`	1.50
	Grammar	⋆	—	`(topleft (intersection (set AllTiles) (coloredTiles Red)))`	4.73
		⚅	—	`(+ (== (color 4A) Red) TRUE)`	0.52
Trial 7 	Human	⋆	How many tiles are occupied by ships?	`(++ (map (lambda x0 (size x0)) (set AllColors)))`	2.48
		⚅	Is there a blue tile at 5E?	`(== (color 5E) Blue)`	0.87
	CodeLlama	⋆	What is the location of one red tile?	`(topleft (coloredTiles Red))`	1.79
		⚅	How many tiles is the red ship?	`(size Red)`	1.58
	GPT-4	⋆	How many tiles is the red ship?	`(size Red)`	1.58
		⚅	How many tiles is the red ship?	`(size Red)`	1.58
	Grammar	⋆	—	`(- (setSize (union (coloredTiles Red) (setDifference (union (unique (union (intersection...`	3.28
		⚅	—	`(== (color 6D) Blue)`	0.98
Trial 8 	Human	⋆	At what location is the bottom right part of the red ship?	`(bottomright (coloredTiles Red))`	2.41
		⚅	Is the red ship horizontal?	`(== (orient Red) H)`	0.85

Board	Model		Question	Program	EIG
	CodeLlama	⋆	What is the location of one blue tile?	`(topleft (coloredTiles Blue))`	2.58
		⚅	At what location is the bottom right part of the red ship?	`(bottomright (coloredTiles Red))`	2.41
	GPT-4	⋆	How many tiles is the blue ship?	`(size Blue)`	1.58
		⚅	How many tiles is the red ship?	`(size Red)`	1.57
	Grammar	⋆	—	`(- (setSize (coloredTiles Blue)) (setSize (setDifference (intersection (coloredTiles (color 3A))...`	3.16
		⚅	—	`(orient Red)`	0.85
Trial 9 	Human	⋆	How many tiles in row 4 are occupied by ships?	`(++ (map (lambda x0 (++ (map (lambda y0 (== (rowL y0) 4)) (coloredTiles x0)))) (set AllColors)))`	1.73
		⚅	Is the blue ship 3 tiles long?	`(== (size Blue) 3)`	0.89
	CodeLlama	⋆	Where is the bottom right tile of the blue ship?	`(bottomright (coloredTiles Blue))`	2.25
		⚅	How many tiles is the red ship?	`(size Red)`	1.54
	GPT-4	⋆	How many tiles is the blue ship?	`(size Blue)`	1.58
		⚅	How many tiles is the red ship?	`(size Red)`	1.54
	Grammar	⋆	—	`(- (setSize (coloredTiles Blue)) (setSize (setDifference (intersection (coloredTiles (color 3A))...`	3.21
		⚅	—	`(== (color 5B) Water)`	0.99
Trial 10 	Human	⋆	What is the location of one blue tile?	`(topleft (coloredTiles Blue))`	3.64
		⚅	How many tiles is the blue ship?	`(size Blue)`	1.12
	CodeLlama	⋆	Where is the bottom right part of the purple ship?	`(bottomright (coloredTiles Purple))`	3.80
		⚅	What is the top left tile of the blue ship?	`(topleft (coloredTiles Blue))`	3.64

Board	Model		Question	Program	EIG
	GPT-4	⋆	Where is the purple ship located?	`(topleft (coloredTiles Purple))`	3.64
		⚅	How many tiles is the red ship?	`(size Red)`	1.12
	Grammar	⋆	—	`(bottomright (setDifference (coloredTiles Blue) (coloredTiles (color 6E))))`	3.93
		⚅	—	`(color 1C)`	1.95
Trial 11 	Human	⋆	What is the location of one purple tile?	`(topleft (coloredTiles Purple))`	3.88
		⚅	How many tiles is the blue ship?	`(size Blue)`	0.97
	CodeLlama	⋆	What is the location of one purple tile?	`(topleft (coloredTiles Purple))`	3.88
		⚅	Is there a ship at 2E?	`(color 2E)`	1.07
	GPT-4	⋆	What is the position of one tile of the red ship?	`(topleft (coloredTiles Red))`	3.88
		⚅	How many tiles is the blue ship?	`(size Blue)`	0.97
	Grammar	⋆	—	`(topleft (coloredTiles Purple))`	3.88
		⚅	—	`(+ (size Red) (rowL 3F))`	1.02

Board	Model		Question	Program	EIG
Trial 12 	Human	⋆	At what location is the top left part of the blue ship?	`(topleft (coloredTiles Blue))`	4.16
		⚅	Is there any part of the blue ship in row 1?	`(any (map (lambda y0 (== (rowL y0) 1)) (coloredTiles Blue)))`	0.97
	CodeLlama	⋆	At what location is the top left part of the blue ship?	`(topleft (coloredTiles Blue))`	4.16
		⚅	How many tiles is the blue ship?	`(size Blue)`	1.47
	GPT-4	⋆	How many tiles is the blue ship?	`(size Blue)`	1.47
		⚅	How many tiles is the blue ship?	`(size Blue)`	1.47
	Grammar	⋆	—	`(topleft (unique (coloredTiles Red)))`	4.16
		⚅	—	`(color 6D)`	1.13
Trial 13 	Human	⋆	At what location is the top left part of the purple ship?	`(topleft (coloredTiles Purple))`	3.99
		⚅	How many tiles is the blue ship?	`(size Blue)`	1.57
	CodeLlama	⋆	At what location is the top left part of the purple ship?	`(topleft (coloredTiles Purple))`	3.99
		⚅	How many tiles is the blue ship?	`(size Blue)`	1.57
	GPT-4	⋆	How many tiles is the blue ship?	`(size Blue)`	1.57
		⚅	How many tiles is the blue ship?	`(size Blue)`	1.57
	Grammar	⋆	—	`(topleft (coloredTiles Purple))`	3.99
		⚅	—	`(setSize (setDifference (coloredTiles (color 1E)) (unique (coloredTiles Blue))))`	2.03
Trial 14 	Human	⋆	What is the location of one red tile?	`(topleft (coloredTiles Red))`	4.00
		⚅	Is there any part of the red ship in column A?	`(any (map (lambda y0 (== (colL y0) 1)) (coloredTiles Red)))`	0.95
	CodeLlama	⋆	At what location is the top left part of the blue ship?	`(topleft (coloredTiles Blue))`	1.58
		⚅	How many tiles is the red ship?	`(size Red)`	1.39

Board	Model		Question	Program	EIG
	GPT-4	⋆	How many tiles is the red ship?	`(size Red)`	1.39
		⚅	Is the blue ship vertical?	`(== (orient Blue) V)`	0.93
	Grammar	⋆	—	`(topleft (coloredTiles Red))`	4.00
		⚅	—	`(topleft (setDifference (unique (union (coloredTiles Water) (set AllTiles))) (unique (union...`	2.71
Trial 15 	Human	⋆	At what location is the top left part of the red ship?	`(topleft (coloredTiles Red))`	4.18
		⚅	Is the red ship 3 tiles long?	`(== (size Red) 3)`	0.83
	CodeLlama	⋆	How many tiles is the red ship?	`(size Red)`	1.27
		⚅	How many tiles is the red ship?	`(size Red)`	1.27
	GPT-4	⋆	How many tiles is the red ship?	`(size Red)`	1.27
		⚅	How many tiles is the blue ship?	`(size Blue)`	0.00
	Grammar	⋆	—	`(topleft (coloredTiles Red))`	4.18
		⚅	—	`(- (setSize (coloredTiles Water)) (colL 2B))`	1.27
Trial 16 	Human	⋆	How many tiles in row 2 are occupied by ships?	`(++ (map (lambda x0 (++ (map (lambda y0 (== (rowL y0) 2)) (coloredTiles x0)))) (set AllColors)))`	1.93
		⚅	Is the red ship horizontal?	`(== (orient Red) H)`	0.86
	CodeLlama	⋆	What is the location of one red tile?	`(topleft (coloredTiles Red))`	1.99
		⚅	How many tiles is the red ship?	`(size Red)`	1.53
	GPT-4	⋆	How many tiles is the red ship?	`(size Red)`	1.53
		⚅	How many tiles is the red ship?	`(size Red)`	1.53

Board	Model		Question	Program	EIG
	Grammar	⋆	—	(- (setSize (coloredTiles Blue)) (setSize (setDifference (intersection (coloredTiles (color 3A))...	3.13
		⚅	—	(setSize (coloredTiles (color 3B)))	2.14
Trial 17 	Human	⋆	How many tiles in row 1 are occupied by ships?	(++ (map (lambda x0 (++ (map (lambda y0 (== (rowL y0) 1)) (coloredTiles x0)))) (set AllColors)))	2.21
		⚅	How many tiles is the purple ship?	(size Purple)	0.92
	CodeLlama	⋆	Where is the top left part of the red ship?	(topleft (coloredTiles Red))	1.92
		⚅	How many tiles is the red ship?	(size Red)	1.52
	GPT-4	⋆	How many tiles is the red ship?	(size Red)	1.52
		⚅	How many tiles is the red ship?	(size Red)	1.52
	Grammar	⋆	—	(setSize (union (coloredTiles (color 6A)) (union (intersection (set AllTiles) (unique (coloredTiles...	2.98
		⚅	—	(color 1C)	0.97
Trial 18 	Human	⋆	At what location is the bottom right part of the purple ship?	(bottomright (coloredTiles Purple))	2.50
		⚅	How many tiles is the purple ship?	(size Purple)	1.56
	CodeLlama	⋆	How many tiles is the purple ship?	(size Purple)	1.56
		⚅	How many tiles is the red ship?	(size Red)	0.00
	GPT-4	⋆	How many tiles is the purple ship?	(size Purple)	1.56
		⚅	How many tiles is the purple ship?	(size Purple)	1.56
	Grammar	⋆	—	(setSize (coloredTiles (color 2B)))	2.50
		⚅	—	(size Purple)	1.56

8.1 Full results

Model	k	EIG μ	EIG σ_M	% Valid μ	% Valid σ_M	% Informative μ	% Informative σ_M	Program Depth μ	Program Depth σ_M	Program Size μ	Program Size σ_M	Question Words μ	Question Words σ_M
Human	1	1.27	0.04	1.00	0.00	0.97	0.01	3.22	0.07	4.51	0.14	7.12	0.08
Grammar	1	0.36	0.00	1.00	0.00	0.38	0.00	3.01	0.00	5.13	0.01	–	–
	5	0.98	0.00	1.00	0.00	0.89	0.00	2.74	0.00	4.07	0.01	–	–
	10	1.25	0.00	1.00	0.00	0.98	0.00	2.82	0.00	4.12	0.01	–	–
	20	1.49	0.00	1.00	0.00	1.00	0.00	3.08	0.01	4.62	0.02	–	–
	50	1.86	0.00	1.00	0.00	1.00	0.00	3.65	0.01	5.71	0.04	–	–
CodeLlama-7b	1	0.65	0.02	0.75	0.01	0.45	0.01	2.64	0.02	3.24	0.04	6.66	0.04
	5	1.24	0.04	0.99	0.01	0.90	0.02	2.49	0.04	2.89	0.08	6.77	0.09
	10	1.55	0.06	0.99	0.01	0.97	0.01	2.36	0.05	2.58	0.09	7.10	0.13
	20	1.83	0.10	1.00	0.00	1.00	0.00	2.34	0.06	2.46	0.11	7.61	0.21
	50	2.31	0.20	1.00	0.00	1.00	0.00	2.58	0.13	2.69	0.22	8.56	0.37
GPT-4 (textual, few-shot)	1	0.77	0.02	0.88	0.01	0.59	0.01	2.61	0.02	3.22	0.04	6.23	0.03
	5	1.16	0.04	0.98	0.01	0.86	0.02	2.47	0.05	2.92	0.09	6.48	0.05
	10	1.43	0.05	1.00	0.00	0.97	0.01	2.33	0.06	2.62	0.12	6.71	0.07
	20	1.65	0.09	1.00	0.00	1.00	0.00	2.17	0.04	2.26	0.06	6.90	0.11
	50	2.04	0.19	1.00	0.00	1.00	0.00	2.19	0.07	2.19	0.07	7.22	0.14
GPT-4 (textual, zero-shot)	1	0.66	0.01	0.40	0.01	0.35	0.01	3.73	0.04	5.04	0.09	5.19	0.02
	5	0.74	0.02	0.60	0.03	0.54	0.03	3.53	0.08	4.82	0.17	5.34	0.04
	10	0.79	0.02	0.77	0.03	0.73	0.03	3.60	0.10	4.89	0.21	5.37	0.06
	20	0.82	0.03	0.92	0.03	0.88	0.03	3.66	0.15	5.05	0.32	5.39	0.07
	50	0.92	0.03	1.00	0.00	1.00	0.00	3.42	0.11	4.33	0.14	5.36	0.11
GPT-4 (grid, few-shot)	1	0.62	0.02	0.85	0.01	0.49	0.01	2.72	0.02	3.42	0.04	6.06	0.03
	5	1.00	0.03	0.96	0.01	0.77	0.02	2.56	0.05	3.10	0.10	6.31	0.06
	10	1.18	0.05	0.99	0.01	0.87	0.03	2.41	0.06	2.80	0.13	6.58	0.08
	20	1.38	0.07	1.00	0.00	0.92	0.03	2.27	0.08	2.50	0.16	6.84	0.12
	50	1.64	0.14	1.00	0.00	1.00	0.00	2.25	0.07	2.39	0.12	7.22	0.24
GPT-4 (grid, zero-shot)	1	0.56	0.01	0.55	0.01	0.39	0.01	3.30	0.03	4.49	0.06	5.85	0.04
	5	0.79	0.02	0.88	0.02	0.80	0.02	3.24	0.05	4.42	0.10	5.71	0.07
	10	0.89	0.02	0.96	0.01	0.93	0.02	3.20	0.06	4.33	0.12	5.82	0.09
	20	0.94	0.01	1.00	0.00	1.00	0.00	3.16	0.08	4.26	0.16	5.86	0.11
	50	0.99	0.02	1.00	0.00	1.00	0.00	3.19	0.15	4.36	0.30	5.83	0.21
GPT-4 (visual, few-shot)	1	0.54	0.01	0.80	0.01	0.46	0.01	3.02	0.02	4.01	0.04	5.69	0.03
	5	0.89	0.03	0.91	0.01	0.75	0.02	2.97	0.06	3.92	0.12	5.92	0.07
	10	1.08	0.04	0.99	0.01	0.92	0.02	2.81	0.07	3.60	0.16	6.16	0.10
	20	1.29	0.06	1.00	0.00	0.98	0.02	2.56	0.09	3.07	0.19	6.56	0.15
	50	1.56	0.12	1.00	0.00	1.00	0.00	2.42	0.13	2.72	0.25	7.03	0.24
GPT-4 (visual, zero-shot)	1	0.34	0.01	0.58	0.01	0.25	0.01	2.18	0.02	2.28	0.03	1.11	0.01
	5	0.73	0.03	0.70	0.02	0.56	0.03	2.18	0.04	2.30	0.07	1.03	0.01
	10	0.88	0.04	0.80	0.03	0.71	0.03	2.07	0.03	2.10	0.04	1.00	0.00
	20	0.99	0.05	1.00	0.00	0.92	0.03	2.07	0.04	2.09	0.05	1.00	0.00
	50	1.19	0.07	1.00	0.00	1.00	0.00	2.22	0.11	2.31	0.15	1.00	0.00
GPT-4 (no board, few-shot)	1	0.60	0.02	0.68	0.01	0.43	0.01	3.08	0.03	4.12	0.07	6.28	0.03
	5	0.98	0.03	0.98	0.01	0.89	0.02	3.01	0.07	3.97	0.13	6.24	0.08
	10	1.19	0.05	1.00	0.00	0.97	0.01	2.82	0.07	3.59	0.15	6.31	0.12
	20	1.38	0.08	1.00	0.00	0.98	0.02	2.73	0.11	3.42	0.24	6.80	0.19
	50	1.75	0.18	1.00	0.00	1.00	0.00	2.72	0.19	3.28	0.40	7.64	0.36
GPT-4 (no board, zero-shot)	1	0.65	0.01	0.69	0.01	0.50	0.01	3.37	0.03	4.67	0.05	6.55	0.02
	5	0.81	0.02	0.94	0.01	0.81	0.02	3.32	0.05	4.59	0.10	6.27	0.04
	10	0.88	0.02	1.00	0.00	0.92	0.02	3.33	0.07	4.61	0.14	6.26	0.06
	20	0.93	0.03	1.00	0.00	0.94	0.02	3.47	0.12	4.89	0.24	6.39	0.10
	50	1.01	0.02	1.00	0.00	1.00	0.00	3.89	0.24	5.72	0.49	6.83	0.19

Table 3: Full statistics for all models and values of k. μ and σ_M denote sample mean and standard error, respectively, and are computed across all board contexts. Questions that translated to a parseable program are considered Valid, and those that achieved EIG > 0 are considered Informative. Program Depth and Size refer to the depth and number of nodes of the program abstract syntax tree. Question Words measures the number of words in the natural language question.

9 Prompts

Our model procedurally constructs few-shot LLM prompts to elicit task-relevant questions and translations. There are two prompt formats: one for question-generation and one for translation. Each prompt is structured as a series of messages conveying instructions, few-shot examples, or information about the target task. In some cases, the format of the message varies depending on the modality of the board representation (textual, grid, or visual).

Following emerging conventions around APIs for conversational AI models, each component is labeled with a *role*. `System` provides general high-level instructions; `User` indicates inputs from a user; and `Assistant` indicates responses generated from the model. These role labels are either passed as metadata (for GPT-4) or prepended to the text of each message (CodeLlama). Note that the purpose of these role labels is to mock illustrate a desired interaction pattern; the LLM only generates text at the end of the conversation.

9.1 Question generation prompt

Instructions The prompt begins with a system message explaining the role of the LLM ("You are a game-playing agent..."). This is followed by a set of general instructions describing the Battleship task. Finally, one of three modality-specific messages is given to describe the format of the board.

Few-shot examples Next, to illustrate the desired behavior, we provide several few-shot examples of boards and questions. Concretely, we randomly choose 3 boards that are not the target board, and randomly choose 10 questions for each board from the human data. (All sampling is done without replacement.) In the "no board" condition, the board representation is omitted, but the example questions are still present. In the zero-shot condition, the entire block beginning with "Here are some examples..." is omitted.

Target board Finally, the prompt concludes with the target board in order to elicit a new question from the LLM. In the "no board" condition, the transition message ("Now, it's your turn...") and the target board are both omitted, so that the prompt effectively reduces to a list of example questions that the LLM extends without any knowledge of the board.

```
System  You are a game-playing agent. Read the game instructions and examples
carefully. Respond with a single question that can be answered with one word.
Do not include any other explanation or prose.
```

```
User  You are playing the board game Battleship. There are three ships on
the board: Red, Blue, and Purple. Ships are oriented either horizontally or
vertically and can be 2, 3, or 4 tiles in length. The board is a 6x6 grid,
with numbered rows 1, 2, 3, 4, 5, 6 and lettered columns A, B, C, D, E, F.
Coordinates are specified as a row, column pair. For example, 2-C is the
tile in row 2, column C.

You will be given a partially-revealed game board. Your task is to ask a
single question that will help you gain information about the position of the
remaining hidden ships on the board. You can ask any question, but it must
be answerable with a single word answer.
```

```
User  (textual)  The
board is represented
as a textual
description.
```

```
User  (grid)  The
board is represented
as a grid with the
following symbols:

H: Hidden
W: Water
R: Red ship
B: Blue ship
P: Purple ship
```

```
User  (visual)  The
board is represented
as an image, with
light gray indicating
hidden tiles, dark
gray indicating water
tiles, and red, blue
and purple indicating
ship tiles.
```

User Here are some examples of questions from other agents about different boards.

3x

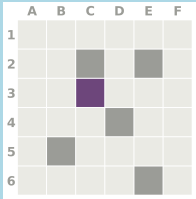
User (textual)

2-C is a water tile.
2-E is a water tile.
3-C is a purple ship tile.
4-D is a water tile.
5-B is a water tile.
6-E is a water tile.

User (grid)

```
  A B C D E F
1 H H H H H H
2 H H W H W H
3 H H P H H H
4 H H H W H H
5 H W H H H H
6 H H H H W H
```

User (visual)



N=10

Assistant At what location is the top left part of the red ship?

Assistant Is the red ship horizontal?

Assistant Is there any ship in column F?

User Now, it's your turn. Here is your board:

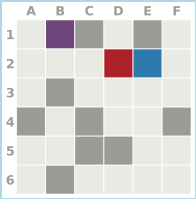
User (textual)

1-B is a purple ship tile.
1-C is a water tile.
1-E is a water tile.
2-D is a red ship tile.
2-E is a blue ship tile.
3-B is a water tile.
...

User (grid)

```
  A B C D E F
1 H P W H W H
2 H H H R B H
3 H W H H H H
4 W H W H H W
5 H H W W H H
6 H W H H H H
```

User (visual)



Assistant ...

9.2 Translation prompt

Instructions The prompt begins with a system message containing general instructions identical to the ones given in the question-generation prompt.

Few-shot examples The body of the prompt consists of 12 pairs of (language, code) examples illustrating the desired translation behavior. These examples are randomly sampled without replacement from the human data. We exclude any examples pertaining to the same board as the one targeted during question-generation.

Target language Finally, the prompt concludes with a target question in language, which the LLM translates into code.

```
System  You are playing the board game Battleship. There are three ships on
the board: Red, Blue, and Purple. Ships are oriented either horizontally or
vertically and can be 2, 3, or 4 tiles in length. The board is a 6x6 grid,
with numbered rows 1, 2, 3, 4, 5, 6 and lettered columns A, B, C, D, E, F.
Coordinates are specified as a row, column pair.  For example, 2-C is the
tile in row 2, column C.

Your task is to translate each of the user's questions into a query program.
```

N=12

```
User  How many tiles is the red ship?

Assistant  (size Red)

User  Do the red ship and the purple ship touch?

Assistant  (touch Red Purple)

User  Is there a ship at 1F?

Assistant  (not (== (color 1F) Water))

User  Is the blue ship horizontal?

Assistant  (== (orient Blue) H)

User  How many ships are horizontal?

Assistant  (++ (map (lambda x0 (== (orient x0) H)) (set AllColors)))
```

```
User  Are there more horizontal ships than vertical ships?

Assistant  ...
```