Shoot First, Ask Questions Later? Building Rational Agents that Explore and Act Like People

Gabriel Grand[1,2*] **Valerio Pepe**[3*] **Joshua B. Tenenbaum**[1,2] **Jacob Andreas**[1]
[1]MIT CSAIL [2]MIT Brain and Cognitive Sciences [3]Harvard SEAS
**Equal contribution*

Abstract

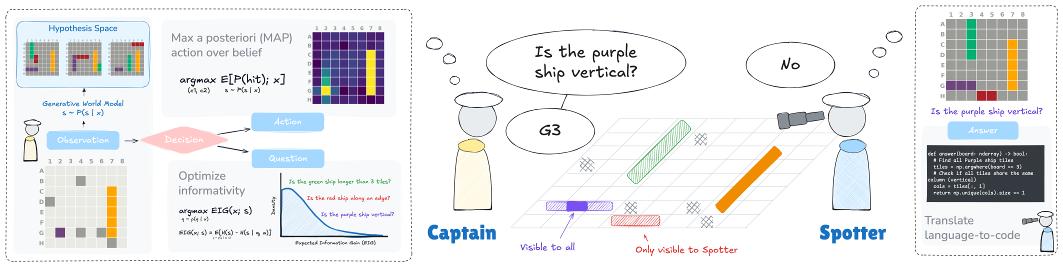
Many emerging applications of AI—from scientific discovery to medical diagnosis—require agents to seek information strategically: forming hypotheses, asking targeted questions, and making decisions under uncertainty. In high-stakes settings with limited resources, do language models (LMs) behave like rational agents? Drawing on insights from human cognition, we develop methods to evaluate and enhance agentic information-seeking. First, we introduce a decision-oriented dialogue task called *Collaborative Battleship*, in which a *Captain* must balance exploration (asking questions) and action (taking shots), while a *Spotter* must supply accurate, contextually-grounded answers. Compared to human players (*N=42*), we find that many LM agents struggle to ask informative questions, produce accurate answers, and identify high-utility actions. To address these gaps, we develop novel Monte Carlo inference strategies for LMs inspired by Bayesian Experimental Design (BED). For Spotter agents, our approach boosts accuracy by up to 14.7% absolute over LM-only baselines; for Captain agents, it raises expected information gain (EIG) by up to 0.227 bits (94.2% of the achievable noise ceiling). Combined, these components yield sharper targeting (+0.303–0.374 F1), and enable weaker LMs, such as Llama-4-Scout, to outperform both humans (8% $\rightarrow$ 82% win rate) and frontier models (0% $\rightarrow$ 67% win rate vs. GPT-5) at $\approx$1% of GPT-5's cost. We replicate these findings on *Guess Who?*, where our methods significantly boost accuracy (+28.3–42.4 p.p.), demonstrating their general applicability for building information-seeking agents.

gabegrand.github.io/battleship

1 Introduction

Language models (LMs) are rapidly evolving from chat-based assistants into fully-fledged agents that interact with the world. Some of the most exciting applications of agents—conducting scientific experiments, conjecturing new mathematical theorems, or discovering novel drugs (Gottweis et al., 2025; Lu et al., 2024; Poesia et al., 2024; Schmidgall et al., 2025)—involve seeking "hits" in combinatorially vast hypothesis spaces. Traditional accounts of information-seeking assume agents are capable of various rational, probabilistic inferences; e.g., inferring belief states, reasoning about uncertainty, and navigating explore/exploit tradeoffs (Anderson, 1990; Auer et al., 2002; Lindley, 1956; MacKay, 1992; Sutton & Barto, 2018). To what extent can current LMs, which are typically optimized to *answer* users' queries (Ouyang et al., 2022; Rafailov et al., 2023), instead *ask good questions* for themselves? And what strategies can we use to improve their information-seeking abilities at inference time?

In this work, we aim to both evaluate and improve the ability of frontier models to ask goal-directed questions and take actions in a dynamic environment. Our setting is an adaptation of the classic board game *Battleship* where players may ask *natural language questions* to gain information about hidden ships. We further extend this paradigm, which was originally developed to study human question-asking (Rothe et al., 2017; 2018; 2019), into a two-player dialogue and decision-making task. We conduct experiments with both human-human and agent-agent pairings, comparing the strategies that LMs employ against both human behavior and idealized resource rational strategies that combine LMs with Bayesian inference techniques.


Hypothesis Space
Generative World Model
$s \sim P(s \mid x)$
Observation
Decision
Action
Question
Max a posteriori (MAP) action over belief
$\arg\max_{(c1, c2)} E_{s \sim P(s \mid x)}[P(hit); x]$
Optimize informativity
$\arg\max_{q \sim p(q \mid x)} EIG(x; s)$
$EIG(x; s) = E_{y \sim p(y \mid x; s)}[H(s) - H(s \mid q, a)]$
Is the green ship longer than 3 tiles?
Is the red ship along an edge?
Is the purple ship vertical?
Density
Expected Information Gain (EIG)
Is the purple ship vertical?
G3
No
Captain
Spotter
Visible to all
Only visible to Spotter
Is the purple ship vertical?
Answer
Translate language-to-code

```
def answer(board: ndarray) -> bool:
  # Find all Purple ship tiles
  tiles = np.argwhere(board == 3)
  # Check if all tiles share the same
column (vertical)
  cols = tiles[:, 1]
  return np.unique(cols).size == 1
```

Figure 1: **Illustration of our *Collaborative Battleship* game.** On each turn, the *Captain* must choose whether to gather information (ask a question) or take action (shoot at a tile). The *Spotter* sees the full board, but can only provide yes/no answers. Each role requires well-defined forms of internal reasoning (thought bubbles), which we implement as Monte Carlo inference over an approximate hypothesis space. This framework allows us to compare both humans and LM agents against idealized Bayesian strategies in a controlled setting.

Our Battleship task tests several distinct cognitive capabilities: (1) *Asking informative questions* that effectively reduce uncertainty; (2) *Providing accurate answers* that are grounded in both the current observation state and the dialogue context; (3) *Taking strategic actions* that leverage available information; (4) *Navigating explore/exploit tradeoffs* in order to balance information-gathering with goal-directed behavior. The minimalistic environment, which shares elements of other challenging text- and grid-based evaluations (Chollet et al., 2024; Guertler et al., 2025; Jansen et al., 2024; Ke et al., 2024; Wang et al., 2022; Yao et al., 2025), provides an ideal testbed for studying Bayesian experimental design (BED; Chaloner & Verdinelli, 1995; Lindley, 1956; Rainforth et al., 2023) in complex state spaces. In particular, questions in Battleship are directly translatable to Python programs; executing these against a sampled "hypothesis space" of game states to compute their expected information gain (EIG) provides a robust way to compare the utility of human and model-generated questions.

As the foundation of our study, we collect 126 full human-human game trajectories (N=42 participants), capturing both dialogue and actions. Our BATTLESHIPQA dataset provides two complementary evaluation settings: **SpotterQA**, which tests grounded answering on 931 gold yes/no questions with expert annotations for answers and various question features, and **CaptainQA**, which tests full strategic gameplay under limited questions and shots.

Evaluating current LMs on these benchmarks reveals a wide spectrum of capability. Weaker models like Llama-4-Scout only marginally exceed random baselines on both SpotterQA (62.2% accuracy) and CaptainQA (68.8% win rate vs. random firing), while frontier reasoning models like GPT-5 match or exceed average human performance. On the answering side, we find that Python code generation substantially improves grounding: across 15 LMs, it boosts SpotterQA accuracy by 14.7% over direct answering and chain-of-thought baselines, with especially large gains for models such as GPT-4.1 (75.2% $\rightarrow$ 90.9%) and Claude 4 Opus (86.8% $\rightarrow$ 94.4%). On the question-asking side, we introduce a simple Bayesian sampling method that substantially improves question quality, raising mean per-question expected information gain (EIG) on CaptainQA by up to 0.227 bits—94.2% of the information-theoretic ceiling. We also observe that many models tend to ask redundant questions that have zero information gain (e.g., Llama-4-Scout: 18.5% of questions; GPT-4o: 14.6%); our method effectively eliminates these cases.

In total, we introduce three rational strategies for question-asking (Bayes-Q), move selection (Bayes-M), and decision-making (Bayes-D). When combined, these strategies yield substantial improvements in overall game performance, as measured by targeting accuracy (+0.397 F1 for Llama-4-Scout, +0.332 F1 for GPT-4o). Remarkably, these Bayesian enhancements enable even weak LMs to reach superhuman-level performance, with win rates of 81–82% against humans and 67% against GPT-5, all while maintaining substantial cost savings (99.7× for Llama-4-Scout and 2.8× for GPT-4o relative to GPT-5). Finally, we replicate our experiments on the *Guess Who?* task from TextArena (Guertler et al., 2025) and observe similarly significant gains (+42.4 p.p. for Llama-4-Scout and +28.3 p.p. for GPT-4o), demonstrating that our approach generalizes beyond the Battleship domain.

In sum, our work provides both practical and theoretical contributions. Concretely, we introduce a reusable evaluation harness for studying agentic information-seeking, and curate a novel multimodal dataset, BATTLESHIPQA, that captures rich pragmatic phenomena in grounded dialogue. Conceptually, we formalize several Bayesian-inspired inference-time strategies that can be applied to other discovery settings to build rational information-seeking agents.

2 The Battleship Game

Our environment draws inspiration from the cognitive science literature, where Battleship-like tasks have previously been utilized to study human information-seeking behavior. In prior work, single-player participants viewed partially-revealed game boards and decided what tiles to reveal (Gureckis & Markant, 2009; Markant & Gureckis, 2012; 2014) or what questions to ask (Rothe et al., 2017; 2018; 2019). Here, we adapt this paradigm to study both humans and language model agents. Our *Collaborative Battleship* game (Fig. 1) is played by two players: a partially-informed *Captain* who must balance exploration (asking questions) and exploitation (taking shots), and a fully-informed *Spotter* who must provide accurate answers that are grounded in both the game state and the ongoing dialogue. Further details about the game rules, interface, and data collection are provided in §A.

Our work extends the *Battleship* paradigm along several key dimensions. (1) *Full multi-turn games:* Prior work was limited to static "snapshots" of game states; here, we simulate full game trajectories. (2) *Dialogue setting:* To enable multi-turn play, we introduce a second player (the Spotter), whose role is necessary to provide real-time answers to the Captain's questions. (3) *Python programs:* In prior work, questions were represented as programs in a hand-engineered domain-specific language (DSL), with translation performed either manually (Rothe et al., 2017) or using LMs (Grand et al., 2024). Here, we represent questions with Python programs, which are both more expressive and easier for LMs to generate. (4) *Information bottleneck:* In prior work, players could in principle ask questions like, "What are the coordinates of all the ships?". To prevent players from asking "game-breaking" questions in order to achieve monetary bonuses (§4.1), we deliberately restrict the Spotter to "Yes" or "No" answers. While less open-ended, such information bottlenecks—also found in games like Twenty Questions, Guess Who, Mastermind, and the original Battleship—ensure strategic balance and enable the study of explore/exploit decision-making.

3 Formal Framework: Bayesian Experimental Design

We cast question selection in our Battleship variant as Bayesian Experimental Design (BED): on each turn we choose to explore—by asking a yes/no question to gain information—or to act—by firing at a hidden tile.

- The hidden board is a random variable $S \in \mathcal{S}$. The observed partial board x induces the feasible set $\mathcal{S}_{\vdash x} = \{\, s \in \mathcal{S} : \ s \text{ is consistent with } x \,\}$.
- The belief at the start of turn t given history $\mathcal{H}_{1:t}$ is $\pi_t(s) = \Pr(S = s \mid x, \mathcal{H}_{1:t})$. Intuitively, this belief places weight on every board that could still be true given what we have seen so far.
- A natural-language question q translates to a deterministic function $f_q : \mathcal{S} \to \{0,1\}$. The true (noise-free) answer is $A_q := f_q(S) \in \{0,1\}$. For the question asked at turn t, we write $A_t := f_{q_t}(S)$.

Observation Model (Noisy Spotter) In our setting, the Spotter plays the role of an observation model. Assuming an oracle Spotter, the predictive probability that the true answer will be "yes" is simply

$$p_t \;:=\; \Pr(A_t = 1 \mid x, \mathcal{H}_{1:t}) \;=\; \sum_{s \in \mathcal{S}_{\vdash x}} \pi_t(s)\, \mathbf{1}\{f_{q_t}(s) = 1\}\,. \tag{1}$$

We expect that the Spotter (human or AI) may occasionally make mistakes, either because they misread the board, misinterpreted the question, or miscommunicated the answer. Accordingly, we model the Spotter (and language-to-code translation) as a *binary symmetric channel* $\mathrm{BSC}(\varepsilon)$ with flip probability $\varepsilon \in [0, \frac{1}{2}]$, yielding a noisy answer $\widetilde{A}_t \in \{0,1\}$.

Bayesian Belief Update. We start with a basic model of the Captain as a Bayesian ideal observer. After asking q_t, the Captain updates their belief by reweighting each candidate board $s \in \mathcal{S}_{\vdash x}$ by how compatible it is with the observed answer $\tilde{a}_t \in \{0,1\}$ under the noise model:

$$\pi_{t+1}(s) \;\propto\; \pi_t(s)\left[(1-\varepsilon)\,\mathbf{1}\{\tilde{a}_t = f_{q_t}(s)\} + \varepsilon\,\mathbf{1}\{\tilde{a}_t \neq f_{q_t}(s)\}\right]. \tag{2}$$

Intuition. Boards that would have produced the observed answer (e.g., $f_{q_t}(s) = \tilde{a}_t$) get *boosted* by a factor $1-\varepsilon$, while boards that would have said the opposite retain some weight ε because mistakes are possible. This gently shifts mass toward boards consistent with what we saw without discarding alternatives outright when $\varepsilon > 0$.

Sequential Monte Carlo (SMC) Approximation. Exact sums over $\mathcal{S}_{\vdash x}$ are typically intractable; we therefore maintain a weighted particle approximation $\{(s_j, w_j^{(t)})\}_{j=1}^{N}$ which is updated via sequential Monte Carlo (SMC; Doucet et al., 2001) with per-turn resampling; see §B for details.

Expected Information Gain. To decide which of many possible candidate questions q_t to ask, we adopt a standard information-theoretic approach (MacKay, 1992; Shannon, 1948) that considers the expected information the Captain will gain about the board from its (noisy) answer:

$$\mathrm{EIG}_\varepsilon(q_t \mid x, \mathcal{H}_{1:t}) \;:=\; I(S; \widetilde{A}_t \mid x, \mathcal{H}_{1:t}) \tag{3}$$

For a $\mathrm{BSC}(\varepsilon)$, this admits a closed form in terms of the binary entropy H_b:

$$\boxed{\mathrm{EIG}_\varepsilon(q_t \mid x, \mathcal{H}_{1:t}) = \mathrm{H}_b\big(\varepsilon + (1-2\varepsilon)\,p_t\big) \;-\; \mathrm{H}_b(\varepsilon), \quad p_t \text{ as in (1)}} \tag{4}$$

Intuition. The first term $\mathrm{H}_b(\varepsilon + (1-2\varepsilon)p_t)$ is the uncertainty in the *noisy* answer we expect to hear; the second term $\mathrm{H}_b(\varepsilon)$ is the uncertainty injected by the channel itself. Their difference is how much uncertainty about the board we expect to remove by asking this question. This quantity is maximized when $p_t \approx \frac{1}{2}$, i.e., when under our current belief the question is about as likely to return "yes" as "no".

3.1 Bayesian Strategies for Exploration and Action

We now describe three strategies that leverage the formal framework above to (i) ask questions that maximize expected information gain, (ii) select actions that maximize hit probability, and (iii) decide between asking a question or taking an action at each turn. These strategies are not necessarily globally optimal, but rather *resource rational* in the sense that they use the current belief π_t to maximize expected utility at each turn.

Asking questions to maximize information gain (Q_{Bayes}). To maximize expected information gain, a simple strategy is to sample a set of candidate questions $\mathcal{Q}$ (e.g., from an LM) and select the one with highest EIG_ε:

$$q_t^\star \in \operatorname*{arg\,max}_{q \in \mathcal{Q}} \; \mathrm{EIG}_\varepsilon(q \mid x, \mathcal{H}_{1:t}), \tag{5}$$

Selecting moves to maximize hit probability (M_{Bayes}). For a candidate tile u (restricted to unrevealed locations), define the probability of a hit under the current belief and the corresponding myopic maximum a posteriori (MAP) action:

$$u_t^\star \in \operatorname*{arg\,max}_{u \text{ unrevealed}} p_t^{\text{hit}}\big(u \mid x, \mathcal{H}_{1:t}\big), \quad p_t^{\text{hit}}\big(u \mid x, \mathcal{H}_{1:t}\big) := \sum_{s \in \mathcal{S}_{\vdash x}} \pi_t(s) \mathbf{1}\{u \text{ contains ship in } s\} \tag{6}$$

Making rational decisions via one-step lookahead (D_{Bayes}). There are many possible strategies for deciding whether to ask a question or take a shot; here, we describe a simple planning-based approach that uses a discounted one-step lookahead to estimate the value of asking a question vs. taking a shot.

Expected post-question hit probability. For a candidate question q_t, we can estimate the next-turn MAP hit probability by marginalizing over possible answers $\tilde{a}$

$$\widehat{p_{t+1}^{\text{hit}}}\big(u \mid x, \mathcal{H}_{1:t}, q_t\big) \;:=\; \sum_{\tilde{a} \in \{0,1\}} \Pr(\widetilde{A}_t = \tilde{a} \mid x, \mathcal{H}_{1:t}, q)\; p_{t+1}^{\text{hit}}\big(u \mid x, \mathcal{H}_{1:t}, \tilde{a}\big). \tag{7}$$

Here $\Pr(\widetilde{A}_t = 1 \mid \cdot) = \varepsilon + (1-2\varepsilon)\,p_t$ with p_t as in (1), and the inner hit probability is evaluated under the updated belief given $\tilde{a}$. Let $\gamma \in [0,1]$ discount the value of information acquired this turn. The strategy proceeds as follows:

1. Select a *potential* question $q_t^\star$ by maximizing EIG via Eq. (5).
2. Compute the pre- and post-question MAP moves $u_t^\star$ and $u_{t+1}^\star$ and corresponding hit probabilities as in Eqs. (6) and (7).
3. If $\gamma\, \widehat{p_{t+1}^{\text{hit}}}(q_t^\star \mid x, \mathcal{H}_{1:t}) \;>\; p_t^{\text{hit}}\big(u_t^\star \mid x, \mathcal{H}_{1:t}\big)$, ask $q_t^\star$; otherwise, act (shoot) at $u_t^\star$.

Intuition. When $\gamma = 1$, the policy asks a question iff the expected one-step improvement in the next-turn MAP hit probability exceeds the current MAP chance; smaller γ prefers acting sooner. While optimizing over longer horizons is possible, the problem of belief-space planning is PSPACE-hard (Papadimitriou & Tsitsiklis, 1987); in practice one-step lookahead is simple and effective.

4 EXPERIMENTS

We present our experimental evaluation in three parts. First, in §4.1, we describe our human behavioral study and dataset, BATTLESHIPQA, in order to build an empirical account of human-like information seeking. Next, we present a series of experiments that evaluate the ability of LMs to answer and ask questions, leveraging BATTLESHIPQA in two distinct ways. In §4.2, we focus on grounded question-answering, evaluating the performance of LMs in answering questions asked by humans. Finally, in §4.3, we focus on question-asking, evaluating the ability of LMs to ask informative questions and make strategic moves in the full game setting.

4.1 EVALUATING HUMAN INFORMATION-SEEKING BEHAVIOR

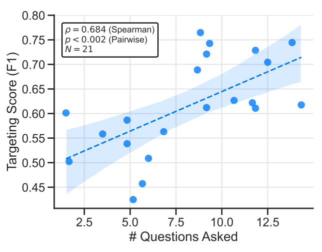
We conducted a two-player, synchronous behavioral study in which participants played *Collaborative Battleship* (described in §2). Participants (N=42) were recruited from Prolific and randomly partnered with another participant. Each pair alternated between the Captain and Spotter roles over 6 games, with a total time commitment of 48-60 minutes and average earnings of \$12.03-\$14.62/hr (including bonuses based on targeting score). In total, we collected data for 18 pre-sampled, 8×8 game boards, allocated evenly across pairs. Further details about the experimental design, implementation, and compensation are provided in §A.2.

Analysis of human data. We manually annotated the human data in order to (1) establish a set of gold labels for evaluating model performance and (2) categorize the types of questions being asked. First, we annotated all questions with a "gold answer" label and filtered out questions for which there was inter-annotator disagreement. After this process, we obtained a gold dataset of 931 questions, establishing a human accuracy baseline of 92.5%.

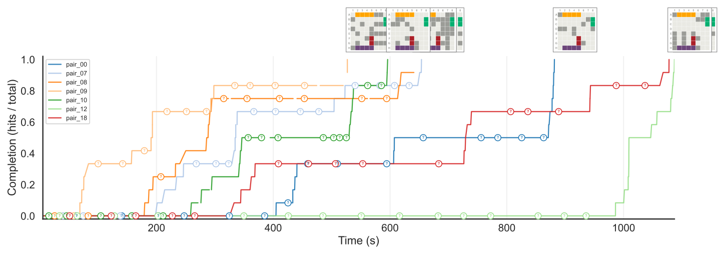
Next, we categorized questions into two groups based on whether they could be answered solely from the true board S (*simple*) or required additional context from the game history $\mathcal{H}_{1:t}$ (*complex*). Within the "complex" category, we futher annotated for various linguistic and pragmatic phenomena, such as *discourse-dependence* (e.g., "Is it longer than 3 tiles?" where "it" references a specific ship), *state-dependence* (e.g., "Should I keep firing up?"), *vagueness* (e.g., "Is there a ship near the center?") and *ambiguity* (i.e., multiple possible interpretations). Further definitions and details are given in §A.3; illustrated examples are provided in §A.4.

4.2 SPOTTERQA: MODELING QUESTION-ANSWERING

We begin our modeling experiments with a focused evaluation of models' performance in the Spotter role, since providing accurate answers is a crucial signal of models' ability to reason about the game state. In particular, we are interested in two answering strategies that we hypothesize should improve

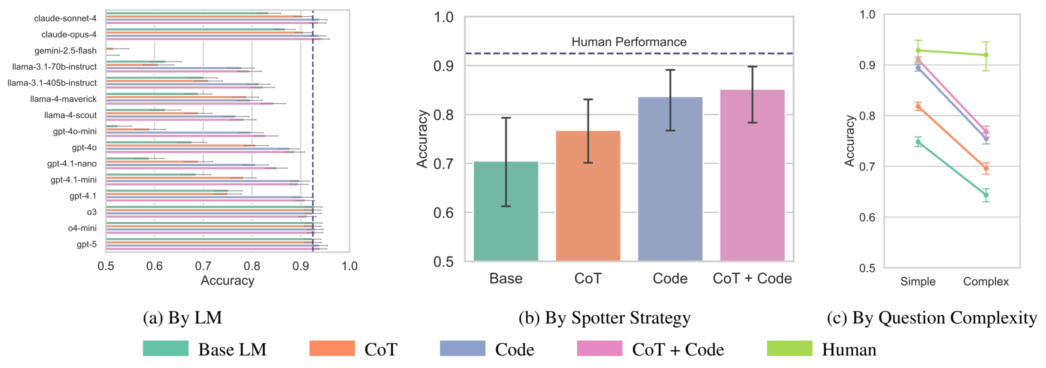


(a) Mean targeting score (F1) vs. # questions asked for each human pair.



(b) Timeline of multiple human-human games of Battleship for a single board (B02). Each ? represents a question asked; colors indicate different players.

Figure 2: **Human results highlights.** (a) Asking questions correlates with performance ($\rho = 0.684, p < 0.002$); (b) Different players demonstrate widely varying explore/exploit strategies; some alternate between asking questions and making moves, while others focus on information-gathering before taking any actions.



(a) By LM (b) By Spotter Strategy (c) By Question Complexity

Figure 3: **SpotterQA key results.** (a) Accuracy varies significantly across the 15 LMs tested. (b) Across LMs, code generation improves accuracy over direct answering and CoT baselines. (c) Whereas human accuracy remains roughly consistent, LMs degrade significantly on "complex" questions that require context. Dashed lines indicate mean human accuracy ($\mu = 92.5\%$) w/r/t gold answer labels. Error bars indicate 95% CIs.

performance: chain-of-thought reasoning (**CoT**; Kojima et al., 2022; Nye et al., 2021; Wei et al., 2022) and language-to-code translation (**Code**; Austin et al., 2021; Wong et al., 2023, see §D.1.1 for examples of synthesized programs). We evaluate these two strategies alongside a combined **CoT + Code** condition and a direct-answer **Base LM** condition.

Evaluation Details Each item in the SpotterQA benchmark is a tuple $(q_t, A_t, x_t, \mathcal{H}_{1:t})$ containing a question, answer, observed board, and history from the human experiment. Boards are represented as Numpy arrays (prompting details in §C.3).[1] Since questions are independent given the history, sequences can be split up for efficient parallel processing. Accordingly, we are able to evaluate a broad (but non-exhaustive) set of frontier LMs from Anthropic (Anthropic, 2025), Google (Comanici & Team, 2025), Meta (Grattafiori et al., 2024; Meta AI, 2025), and OpenAI (OpenAI, 2024; 2025a;b;c). We query all models via the OpenRouter API with default parameters.

4.2.1 Results

Fig. 3 summarizes the results of our SpotterQA evaluation; full details are provided in §D.1. We highlight three key findings:

Question-answering abilities vary widely across models. As seen in Fig. 3a, accuracy varies widely across the 15 LMs tested, ranging from near-chance (52.5%; GPT-4o-mini) to 92.8% (o3 mini). Notably, several models (o3, o4-mini, and GPT-5) meet or exceed mean human performance (92.5%). (See Table 3 for full results.)

Code generation consistently improves answering accuracy. Across models (Fig. 3b), we find that code improves SpotterQA by 13.2% absolute accuracy points over Base; combining code generation with chain-of-thought (CoT + Code) yields even greater 14.7% gains. A two-sided Mann-Whitney U test confirms that these differences are significant ($p < 0.001$; see Fig. 17). Importantly, for many models, code generation substantially closes the gap between LMs and human performance: for instance, Claude 4 Opus improves from 86.8% (Base) to 94.4% (CoT + Code), and GPT-4.1 improves from 75.2% (Base) to 90.9% (CoT + Code). These results demonstrate that code generation helps LMs to produce more accurate, grounded answers.

LMs struggle with context-dependent questions. As shown in Fig. 3c, whereas human accuracy remains roughly consistent across simple (92.8%) and complex (91.9%) questions, LMs degrade significantly on complex questions (as defined in §4.1). For instance, GPT-4o's accuracy drops from 72.8% on simple questions to 60.4% on complex ones; similarly, Llama-4-Scout drops from 68.0% to 54.0%. Code generation partially mitigates this gap, but even the best model (o3) still falls short of human performance on complex questions (87.4% vs. 91.9%). These results suggest that current LMs still struggle with pragmatic reasoning and context-dependent meanings.

[1]Prior evaluations find that vision language models (VLMs) struggle with spatial reasoning on 2D grids, including Battleship boards (Grand et al., 2024). Accordingly, we use serialized representations, following the predominant approach on ARC-like tasks (Akyürek et al., 2024; Li et al., 2024; Wang et al., 2024).

4.3 CaptainQA: Modeling Question-Asking

In this section, we evaluate the ability of LMs to play the Captain role in the full game setting. We compare a variety of Captain strategies that differ in how they select moves, ask questions, and decide whether to ask or shoot (as formalized in §3.1). A summary is provided in Table 1.

Captain	Decision (D)	Question (Q)	Move (M)
Random	Move	–	$i, j \sim \mathrm{Unif}(\{1, \dots, 8\})$
Greedy	Move	–	$\arg\max_{i,j} \pi_t(\cdot \mid x)$
LM	$p_{\mathrm{LM}}(\cdot \mid x, \mathcal{H}_{1:t})$	$p_{\mathrm{LM}}(\cdot \mid x, \mathcal{H}_{1:t})$	$p_{\mathrm{LM}}(\cdot \mid x, \mathcal{H}_{1:t})$
+ Bayes-Q	$\vdots$	$\arg\max_{q \in \mathcal{Q}} \mathrm{EIG}_{\varepsilon}(q \mid x, \mathcal{H}_{1:t})$	$\vdots$
+ Bayes-M	$\vdots$	$p_{\mathrm{LM}}(\cdot \mid x, \mathcal{H}_{1:t})$	$\arg\max_{i,j} \pi_t(\cdot \mid x, \mathcal{H}_{1:t})$
+ Bayes-QM	$\vdots$	Q_{Bayes}	M_{Bayes}
+ Bayes-QMD	$p^{\text{hit}}(\pi_t) > \gamma p^{\text{hit}}(\pi_{t+1} \mid q_t, \widetilde{A}_t)$	$\vdots$	$\vdots$

Table 1: Summary of Captain strategies. *Random* and *Greedy* are move-only baselines; *LM* is a pure language model, which the *Bayes* strategies build upon. Triple-dots indicates inheritance from the row above.

Experimental Details We evaluate each Captain strategy over 54 games (the 18 pre-sampled boards from the human experiment; §4.1, with 3 random seeds per board). Each game enforces the same constraints as the human setting: a maximum of 15 questions and 40 moves. To control for answer quality, we fix the Spotter to GPT-5 (CoT + Code) for all strategies and set $\varepsilon = 0.1$, calibrated from GPT-5's SpotterQA accuracy (§4.2). Because each turn involves multiple API queries which must be made sequentially, we restrict CaptainQA evaluation to three representative LMs: Llama-4-Scout (small, non-reasoning), GPT-4o (large, non-reasoning), and GPT-5 (large, reasoning-capable). We evaluate every combination of LM and Captain strategy, with the exception of Bayes-QMD, which we were not able to evaluate with GPT-5 due to cost (see Table 5 for a breakdown). For D_{Bayes}, we set $\gamma = 0.95$ to encode a minor discount for future action. For strategies that use LMs, the prompt includes the current board state and the full game history (see §C.2 for prompting details).

Captain Evaluation Metrics We evaluate Captains using five primary metrics that capture different aspects of performance:

- **Targeting Score (F1)**: Overall metric of ship-sinking performance, balancing both precision and recall. This metric treats the board as a binary classification task, where ship tiles correspond to the positive class and water tiles correspond to the negative class.
- **Move Count**: The average number of shots taken per game; equivalently, the average game length (max 40 moves; questions do not count towards the move count).
- **Questions Asked**: The average number of questions asked per game (max 15).
- **Win Rate**: A pairwise metric based on board-matched performance in simulated head-to-head play. For each pair of Captains on the same board, the winner is the one that sinks all ships in the fewest moves, with tiebreaking based on targeting score (F1). Since Captains do not play each other directly, this comparison is computed post-hoc over all pairs of games, averaging over boards.
- **EIG**: The average expected information gain of questions asked. We use $\varepsilon = 0.1$, meaning that the maximum possible EIG is $1 - \mathrm{H}_b(0.1) = 0.531$ bits.

4.3.1 Results

Fig. 4 summarizes the results of our CaptainQA evaluation; full details are provided in §D.2. We highlight four key findings:

Incorporating Bayesian strategies brings weaker models up to super-human performance. As seen in Fig. 4a, the LM-only strategy shows a wide range of performance, with Llama-4-Scout achieving only 0.367 F1, while GPT-5 reaches 0.716. However, adding Bayesian question, move, and decision strategies (+Bayes-QMD) significantly improves performance across all models (e.g., Llama-4-Scout jumps 0.367 → 0.764 F1, GPT-4o 0.450 → 0.782 F1). Remarkably, with the full Bayesian model, both Llama-4-Scout and GPT-4o outperform both humans (0.82–0.83 win rate vs. humans) as well as the strongest LM (0.67 win rate vs. GPT-5), suggesting that Bayesian strategies can compensate for weaker LM capabilities (see win rates in Fig. 18). The LMs equipped with this full Bayesian model also outperform GPT-5 at a fraction of its cost (≈1% of GPT-5's cost for Llama-4-Scout, and ≈35% for GPT-4o, see Table 5) Notably, GPT-5 itself does not significantly benefit from Bayesian question or move selection, indicating that it may already be employing effective versions of these strategies internally.

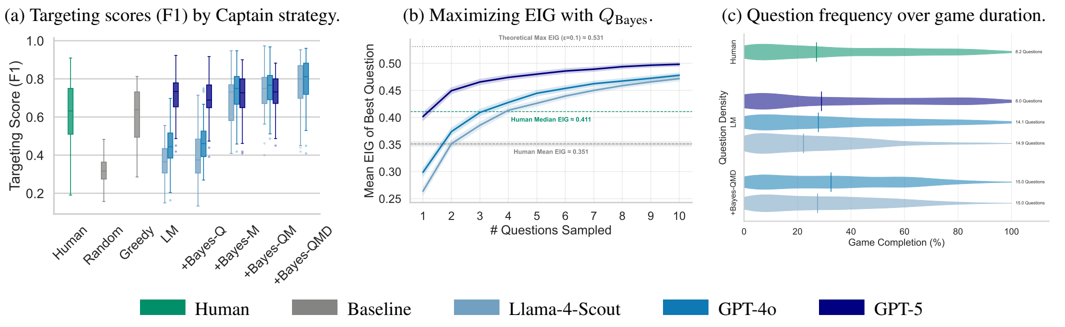


Figure 4: **CaptainQA key results.** (a) Incorporating Bayesian strategies for questions (Q_{Bayes}), moves (M_{Bayes}), and decisions (D_{Bayes}) brings weaker LMs from near-random performance to super-human levels. (b) Sampling up to 10 questions with Q_{Bayes} yields higher EIG. (c) D_{Bayes} helps Llama-4-Scout and GPT-4o to spread out questions over time, more closely matching the behavior of humans and GPT-5.

Inference scaling yields more informative questions for all models. As shown in Fig. 4b, EIG scales with the number of candidate questions sampled under the Q_{Bayes} strategy. Across models, we see EIG improvements of up to 0.227 bits per question (up to 94.2% of the info-theoretic ceiling). Additionally, Q_{Bayes} significantly reduces the proportion of *redundant* questions (EIG = 0; Table 4) asked by Llama-4-Scout (18.5% $\rightarrow$ 0.2%) and GPT-4o (14.6% $\rightarrow$ 1.2%). (We find that both humans and GPT-5 rarely ask redundant questions.) These findings illustrate how LMs—including models like GPT-5 that already perform extensive test-time reasoning—can further benefit from inference scaling techniques designed to improve question quality.

Asking high-EIG questions is not sufficient to guarantee strong game performance. While Q_{Bayes} consistently improves EIG, this does not consistently translate to improved game performance. For instance, both Llama-4-Scout and GPT-4o see significant EIG gains with Q_{Bayes}, but their targeting scores improve only marginally (+0.021–0.026 F1). This suggests that weaker models may not be able to effectively leverage information into accurate moves. In contrast, M_{Bayes}, which explicitly marginalizes over the implications of each question to compute π_t, provides a more reliable mechanism for doing so.

Skilled players ask *some* questions first, *but not all*. As seen in Fig. 4c, both humans and GPT-5 tend to ask several questions early in the game, but also reserve some questions for later turns. In contrast, weaker players (e.g., Llama-4-Scout) tend to front-load all 15 questions—this myopic behavior is mitigated by introducing one-step lookahead (D_{Bayes}), which leads to a more balanced approach. Interestingly, stronger players (humans, GPT-5) ask *fewer* questions overall (8.0–8.2 vs. 14.1–14.9), indicating that they both gain more information per question (higher EIG; Table 4) and make more effective use of that information in their moves (higher F1; Fig. 4a).

5 Generalized Information-Seeking Games

We extend our Bayesian strategies to a general family of information-seeking games from TextArena (Guertler et al., 2025). Here, we present results from the board game "*Guess Who?*" (see §E for full details). This game provides a distinct testbed from *Battleship*, as it involves richer, object-relational semantics and requires more complex reasoning about entities and attributes (e.g., age, clothing, facial hair, etc.). As shown in Fig. 5, both GPT-4o and Llama-4-Scout's success rates improve significantly with QM_{Bayes} (GPT-4o: 0.617 $\rightarrow$ 0.900; Llama-4-Scout: 0.300 $\rightarrow$ 0.724). These results suggest our framework successfully generalizes to information-seeking environments with combinatorial hypothesis spaces.

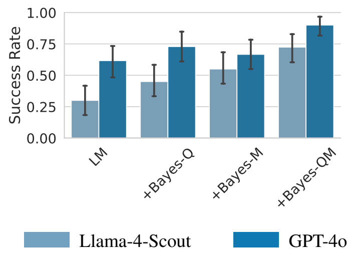


Figure 5: ***Guess Who?*** Our Bayesian strategies yield consistent improvements, replicating the core findings from *Battleship*.

6 Related Work, Discussion, and Conclusion

Expressing queries and hypotheses with programs. Several notable prior works use language-to-code to support efficient probabilistic inferences (Ellis, 2023; Li et al., 2024; Piriyakulkij et al., 2024; Wang et al., 2024; Wong et al., 2023). Our work integrates hypothesis-driven reasoning with other recent approaches for planning and rational agent behavior (Chiu et al., 2023; Curtis et al., 2025; Ying et al., 2024; 2025).

Eliciting user preferences and clarifying ambiguity. Asking informative questions is critical for many user-facing applications of LMs; recent approaches consider structured prompting (Li et al., 2023), entropy reduction (GX-Chen et al., 2025; Hu et al., 2024; Mazzaccara et al., 2024; Piriyakulkij et al., 2023; Rao & Daumé III, 2018; Yu et al., 2020; Zhang & Choi, 2023), Bayesian inference (Handa et al., 2024; Qiu et al., 2025), and constraint satisfaction (Li et al., 2025).

Human information-seeking and resource rationality. Our work is broadly informed by "resource rational" accounts that describe how human behavior is shaped by cognitive constraints (Anderson, 1990; Chater & Oaksford, 1999; Icard, 2025; Leider et al., 2025; Lieder & Griffiths, 2020). In information-seeking tasks, both children and adults greedy heuristics (Markant et al., 2016; Meder et al., 2019; Ruggeri et al., 2016), consider only a few hypotheses at a time (Klayman & Ha, 1989; Vul et al., 2014), and prefer queries that yield easily interpretable information (Cheyette et al., 2023). These ideas motivate our modeling approach, which emphasizes greedy, sample-based strategies over exact planning and inference.

6.1 Limitations and Future Work

Collaborative Battleship gives rise to many rich pragmatic behaviors. While discussed in §§ A.3 and A.4, in general, these are not explicitly modeled; incorporating techniques based on the rational speech acts (RSA) framework (Frank & Goodman, 2012; Hawkins et al., 2017; 2023) could yield agents capable of more sophisticated pragmatic reasoning. In particular, people are highly sensitive to the reliability of information (Harris & Corriveau, 2011; Sperber et al., 2010), as reflected in interactions between human players (e.g., Figs. 14 and 15). In place of a fixed ε, a more robust approach would be to *infer* ε to account for the differences in reliability across individual Spotters. Analogously, in scientific settings, agents may need to adapt to different levels of aleatoric uncertainty (Hüllermeier & Waegeman, 2021).

Our Bayesian strategies rely on the ability to efficiently draw conditional samples $s \sim p(s \mid x, \mathcal{H})$ from a generative "world model." While implementable by hand in a domain like Battleship, in more general settings, we may wish to *learn* a generative model of world states represented as code (e.g., via model synthesis architectures (MSA); Wong et al., 2023; 2025) or images; (e.g., via VAEs or diffusion; Alonso et al., 2024; Ha & Schmidhuber, 2018). Finally, building agents that collaborate effectively with people is increasingly important (Boiko et al., 2023; Noti et al., 2025); *Collaborative Battleship* provides an ideal setting for studying human-agent interactions.

6.2 Conclusion

In this work, we presented a comparative evaluation of human and agent information-seeking. We introduced a cognitively-inspired *Collaborative Battleship* task designed to replicate the core components of Bayesian Experimental Design (BED), including forming hypotheses about latent variables, asking and answering questions, and leveraging context-dependent information to inform actions. Our behavioral study provides a rich, multimodal dataset of human interactions, BattleshipQA, enabling systematic comparison with model performance.

Our results highlight both progress and gaps: while smaller models like Llama-4-Scout and GPT-4o struggle to explore and act coherently, larger reasoning models like GPT-5 approach or even surpass human-level performance at steep resource costs. Notably, humans themselves are not Bayes-optimal reasoners; in general they do not ask maximally-informative questions or make exhaustive use of all available resources. Nevertheless, people are remarkably good at asking useful questions, providing contextually-grounded answers, and making well-informed guesses. Prioritizing strategies that are resource rational is therefore essential if we want to build agents that scale beyond benchmarks to collaborate with people on real-world problems.

Author Contributions

Gabriel Grand* Research conceptualization, narrative development, modeling (design and implementation), human experiments (design and supervision), analysis (Captain and Spotter experiments), figure-making, writing, math and formalization.

Valerio Pepe* Research conceptualization, narrative development, modeling (design and implementation), human experiments (design, implementation, and execution), analysis (Captain and Spotter experiments), writing, TextArena experiments.

Joshua B. Tenenbaum Senior mentorship, research conceptualization, narrative development, human experiments (design and supervision).

Jacob Andreas Senior mentorship, research conceptualization, narrative development, human experiments (design and supervision), analysis of results.

**Equal contribution, co-first authorship.*

Acknowledgments

We thank Brenden Lake, Tomer Ullman, Robert Hawkins, Judy Fan, Lionel Wong, Tan Zhi-Xuan, Belinda Li, Alexis Ross, Sam Cheyette, Guy Davidson, and Graham Todd for helpful discussions and feedback.

The authors gratefully acknowledge support from the Siegel Family Quest for Intelligence, the MIT-IBM Watson AI Lab, the FinTechAI@CSAIL initiative, the Intel Corporation, AFOSR, DARPA, ONR, and the National Science Foundation (NSF) under grants CCF-2217064 and IIS-2238240. G.G. is additionally supported by a NSF Graduate Research Fellowship under Grant No. 2141064. J.B.T. is additionally supported by AFOSR, ONR Science of AI, and Siegel Family Endowment. J.A. is additionally supported by a Sloan Research Fellowship. 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.

Ethics Statement

We acknowledge that our research involves human participants and has been conducted with the utmost consideration for ethical standards. The study protocol was reviewed and approved by the MIT Committee on the Use of Humans as Experimental Subjects (COUHES), which serves as the Institutional Review Board (IRB) at our institution, to ensure compliance with the Ethical Principles and Guidelines for the Protection of Human Subjects of Research, including the Belmont Report principles of respect for persons, beneficence, and justice. All participants provided informed consent, and their privacy was protected throughout the study. We are committed to ensuring that our findings are used responsibly and for the benefit of society.

Reproducibility Statement

We strive to make our research reproducible by providing detailed descriptions of our methods, datasets, and experimental procedures. We encourage other researchers to replicate our findings and build upon our work. All code and data used in our experiments, including the full BattleshipQA dataset, are available at `gabegrand.github.io/battleship`.

Use of AI Assistance

We acknowledge the use of AI tools in the preparation of this manuscript. Specifically, we utilized language models (LMs) to assist with grammar and style editing, literature review, preparation of figures, general coding assistance, and other tasks. We have reviewed and edited the content to ensure accuracy and clarity, and we take full responsibility for the final version of the manuscript.

References

Ekin Akyürek, Mehul Damani, Adam Zweiger, Linlu Qiu, Han Guo, Jyothish Pari, Yoon Kim, and Jacob Andreas. The Surprising Effectiveness of Test-Time Training for Few-Shot Learning, 2024. URL https://arxiv.org/abs/2411.07279.

Abdullah Almaatouq, Joshua Becker, James P. Houghton, Nicolas Paton, Duncan J. Watts, and Mark E. Whiting. Empirica: a virtual lab for high-throughput macro-level experiments. *Behavior Research Methods*, 53(5):2158–2171, 2021. ISSN 1554-3528. doi: 10.3758/s13428-020-01535-9. URL https://doi.org/10.3758/s13428-020-01535-9.

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos J. Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. In Amir Globerson, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/6bdde0373d53d4a501249547084bed43-Abstract-Conference.html.

John R. Anderson. *The adaptive character of thought*. Lawrence Erlbaum, 1990.

Anthropic. Claude opus 4 & claude sonnet 4 — system card, 2025. URL https://www.anthropic.com/claude-4-system-card.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002. doi: 10.1023/A:1013689704352.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program Synthesis with Large Language Models, 2021. URL https://arxiv.org/abs/2108.07732.

Daniil A. Boiko, Robert MacKnight, Ben Kline, and Gabe Gomes. Autonomous chemical research with large language models. *Nature*, 624(7992):570–578, 2023. ISSN 1476-4687. doi: 10.1038/s41586-023-06792-0. URL https://www.nature.com/articles/s41586-023-06792-0. Publisher: Nature Publishing Group.

Kathryn Chaloner and Isabella Verdinelli. Bayesian Experimental Design: A Review. *Statistical Science*, 10(3), 1995. ISSN 0883-4237. doi: 10.1214/ss/1177009939. URL https://projecteuclid.org/journals/statistical-science/volume-10/issue-3/Bayesian-Experimental-Design-A-Review/10.1214/ss/1177009939.full.

N. Chater and M. Oaksford. Ten years of the rational analysis of cognition. *Trends in Cognitive Sciences*, 3(2):57–65, 1999. ISSN 1879-307X. doi: 10.1016/s1364-6613(98)01273-x.

Samuel J. Cheyette, Frederick Callaway, Neil R. Bramley, Jonathan D. Nelson, and Josh Tenenbaum. People seek easily interpretable information. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 45(45), 2023. URL https://escholarship.org/uc/item/5sm2b484.

Justin Chiu, Wenting Zhao, Derek Chen, Saujas Vaduguru, Alexander Rush, and Daniel Fried. Symbolic planning and code generation for grounded dialogue. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 7426–7436, Singapore, 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.460. URL https://aclanthology.org/2023.emnlp-main.460.

Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. ARC Prize 2024: Technical Report, 2024. URL https://arxiv.org/abs/2412.04604.

Gabriel Comanici and the Gemini Team. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *ArXiv preprint*, abs/2507.06261, 2025. URL https://arxiv.org/abs/2507.06261.

Aidan Curtis, Hao Tang, Thiago Veloso, Kevin Ellis, Joshua Tenenbaum, Tomás Lozano-Pérez, and Leslie Pack Kaelbling. LLM-Guided Probabilistic Program Induction for POMDP Model Estimation. *ArXiv preprint*, abs/2505.02216, 2025. URL https://arxiv.org/abs/2505.02216.

Arnaud Doucet, Nando de Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Springer Science & Business Media, 2001. ISBN 978-0-387-95146-1.

Kevin Ellis. Human-like few-shot learning via bayesian reasoning over natural language. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/2aa9b18b9ab37b0ab1fdaae46fb781d4-Abstract-Conference.html.

Michael C. Frank and Noah D. Goodman. Predicting Pragmatic Reasoning in Language Games. *Science*, 336(6084):998–998, 2012. doi: 10.1126/science.1218633. URL https://www.science.org/doi/10.1126/science.1218633. Publisher: American Association for the Advancement of Science.

Juraj Gottweis, Wei-Hung Weng, Alexander Daryin, Tao Tu, Anil Palepu, Petar Sirkovic, Artiom Myaskovsky, Felix Weissenberger, Keran Rong, Ryutaro Tanno, Khaled Saab, Dan Popovici, Jacob Blum, Fan Zhang, Katherine Chou, Avinatan Hassidim, Burak Gokturk, Amin Vahdat, Pushmeet Kohli, Yossi Matias, Andrew Carroll, Kavita Kulkarni, Nenad Tomasev, Yuan Guan, Vikram Dhillon, Eeshit Dhaval Vaishnav, Byron Lee, Tiago R. D. Costa, José R. Penadés, Gary Peltz, Yunhan Xu, Annalisa Pawlosky, Alan Karthikesalingam, and Vivek Natarajan. Towards an AI co-scientist, 2025. URL https://arxiv.org/abs/2502.18864.

Gabriel Grand, Valerio Pepe, Jacob Andreas, and Joshua B. Tenenbaum. Loose LIPS Sink Ships: Asking Questions in Battleship with Language-Informed Program Sampling, 2024. URL https://arxiv.org/abs/2402.19471.

Andrew Grattafiori, Abhimanyu Dubey, and others. The llama 3 herd of models. *ArXiv preprint*, abs/2407.21783, 2024. URL https://arxiv.org/abs/2407.21783.

Leon Guertler, Bobby Cheng, Simon Yu, Bo Liu, Leshem Choshen, and Cheston Tan. TextArena, 2025. URL https://arxiv.org/abs/2504.11442.

Todd Gureckis and Doug Markant. Active Learning Strategies in a Spatial Concept Learning Game. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 31(31), 2009. URL https://escholarship.org/uc/item/55b5m116.

Anthony GX-Chen, Dongyan Lin, Mandana Samiei, Doina Precup, Blake A. Richards, Rob Fergus, and Kenneth Marino. Language Agents Mirror Human Causal Reasoning Biases. How Can We Help Them Think Like Scientists?, 2025. URL https://arxiv.org/abs/2505.09614.

David Ha and Jürgen Schmidhuber. World Models. *ArXiv preprint*, abs/1803.10122, 2018. URL https://arxiv.org/abs/1803.10122.

Kunal Handa, Yarin Gal, Ellie Pavlick, Noah Goodman, Jacob Andreas, Alex Tamkin, and Belinda Z. Li. Bayesian Preference Elicitation with Language Models, 2024. URL https://arxiv.org/abs/2403.05534.

Paul L. Harris and Kathleen H. Corriveau. Young children's selective trust in informants. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 366(1567):1179–1187, 2011. ISSN 0962-8436. doi: 10.1098/rstb.2010.0321. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC3049091/.

Robert D. Hawkins, Mike Frank, and Noah D. Goodman. Convention-formation in iterated reference games. *Cognitive Science*, 2017. URL https://www.semanticscholar.org/paper/Convention-formation-in-iterated-reference-games-Hawkins-Frank/246967b26e0911379122d667f97578834cbbb394.

Robert D. Hawkins, Michael Franke, Michael C. Frank, Adele E. Goldberg, Kenny Smith, Thomas L. Griffiths, and Noah D. Goodman. From partners to populations: A hierarchical Bayesian account of coordination and convention. *Psychological Review*, 130(4):977–1016, 2023. ISSN 1939-1471, 0033-295X. doi: 10.1037/rev0000348. URL https://doi.apa.org/doi/10.1037/rev0000348.

Zhiyuan Hu, Chumin Liu, Xidong Feng, Yilun Zhao, See-Kiong Ng, Anh Tuan Luu, Junxian He, Pang Wei Koh, and Bryan Hooi. Uncertainty of Thoughts: Uncertainty-Aware Planning Enhances Information Seeking in Large Language Models, 2024. URL https://arxiv.org/abs/2402.03271.

Eyke Hüllermeier and Willem Waegeman. Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods. *Machine Learning*, 110(3):457–506, 2021. ISSN 1573-0565. doi: 10.1007/s10994-021-05946-3. URL https://doi.org/10.1007/s10994-021-05946-3.

Thomas F. Icard. Resource Rationality, 2025. URL https://philarchive.org/rec/ICARRT.

Peter A. Jansen, Marc-Alexandre Côté, Tushar Khot, Erin Bransom, Bhavana Dalvi Mishra, Bodhisattwa Prasad Majumder, Oyvind Tafjord, and Peter Clark. Discoveryworld: A virtual environment for developing and evaluating automated scientific discovery agents. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/13836f251823945316ae067350a5c366-Abstract-Datasets_and_Benchmarks_Track.html.

Nan Rosemary Ke, Danny P. Sawyer, Hubert Soyer, Martin Engelcke, David P. Reichert, Drew A. Hudson, John Reid, Alexander Lerchner, Danilo Jimenez Rezende, Timothy P. Lillicrap, Michael Mozer, and Jane X. Wang. Can foundation models actively gather information in interactive environments to test hypotheses?, 2024. URL https://arxiv.org/abs/2412.06438.

Joshua Klayman and Young-won Ha. Hypothesis testing in rule discovery: Strategy, structure, and content. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 15(4):596, 1989. Publisher: American Psychological Association.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

Falk Leider, Frederick Callaway, and Thomas L. Griffiths. The Rational Use of Cognitive Resources, 2025. URL https://www.researchgate.net/publication/391601646_The_Rational_Use_of_Cognitive_Resources.

Belinda Z. Li, Alex Tamkin, Noah Goodman, and Jacob Andreas. Eliciting Human Preferences with Language Models, 2023. URL https://arxiv.org/abs/2310.11589.

Belinda Z. Li, Been Kim, and Zi Wang. QuestBench: Can LLMs ask the right question to acquire information in reasoning tasks? *ArXiv preprint*, abs/2503.22674, 2025. URL https://arxiv.org/abs/2503.22674.

Wen-Ding Li, Keya Hu, Carter Larsen, Yuqing Wu, Simon Alford, Caleb Woo, Spencer M. Dunn, Hao Tang, Michelangelo Naim, Dat Nguyen, Wei-Long Zheng, Zenna Tavares, Yewen Pu, and Kevin Ellis. Combining Induction and Transduction for Abstract Reasoning, 2024. URL https://arxiv.org/abs/2411.02272.

Falk Lieder and Thomas L. Griffiths. Resource-rational analysis: Understanding human cognition as the optimal use of limited computational resources. *Behavioral and Brain Sciences*, 43:e1, 2020. ISSN 0140-525X, 1469-1825. doi: 10.1017/S0140525X1900061X. URL https://www.cambridge.org/core/product/identifier/S0140525X1900061X/type/journal_article.

Dennis V. Lindley. On a measure of the information provided by an experiment. *Annals of Mathematical Statistics*, 27:986–1005, 1956. doi: 10.1214/aoms/1177728069.

Chris Lu, Cong Lu, Robert Tjarko Lange, Jakob Foerster, Jeff Clune, and David Ha. The AI Scientist: Towards Fully Automated Open-Ended Scientific Discovery, 2024. URL https://arxiv.org/abs/2408.06292.

David JC MacKay. Information-based objective functions for active data selection. *Neural Computation*, 4:590–604, 1992. doi: 10.1162/neco.1992.4.4.590.

Bronislaw Malinowski. *The meaning of meaning: a study of the influence of language upon thought and of the science of symbolism*, volume 29. K. Paul, Trench, Trubner & Company, Limited, 1927.

Doug Markant and Todd Gureckis. Does the utility of information influence sampling behavior? *Proceedings of the Annual Meeting of the Cognitive Science Society*, 34(34), 2012. URL https://escholarship.org/uc/item/0fz3d3mw.

Doug Markant and Todd Gureckis. A preference for the unpredictable over the informative during self-directed learning. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 36 (36), 2014. URL https://escholarship.org/uc/item/0059t11p.

Douglas B Markant, Burr Settles, and Todd M Gureckis. Self-directed learning favors local, rather than global, uncertainty. *Cognitive science*, 40(1):100–120, 2016. Publisher: Wiley Online Library.

Davide Mazzaccara, Alberto Testoni, and Raffaella Bernardi. Learning to Ask Informative Questions: Enhancing LLMs with Preference Optimization and Expected Information Gain. *ArXiv preprint*, abs/2406.17453, 2024. URL https://arxiv.org/abs/2406.17453.

Björn Meder, Jonathan D Nelson, Matt Jones, and Azzurra Ruggeri. Stepwise versus globally optimal search in children and adults. *Cognition*, 191:103965, 2019. Publisher: Elsevier.

Meta AI. The Llama 4 herd: The beginning of a new era of natively multimodal AI innovation, 2025. URL https://ai.meta.com/blog/llama-4-multimodal-intelligence/.

Gali Noti, Kate Donahue, Jon Kleinberg, and Sigal Oren. AI-Assisted Decision Making with Human Learning, 2025. URL https://arxiv.org/abs/2502.13062.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. Show Your Work: Scratchpads for Intermediate Computation with Language Models, 2021. URL https://arxiv.org/abs/2112.00114.

OpenAI. GPT-4o system card. *ArXiv preprint*, abs/2410.21276, 2024. URL https://arxiv.org/abs/2410.21276.

OpenAI. Introducing GPT-4.1 in the API, 2025a. URL https://openai.com/index/gpt-4-1/.

OpenAI. GPT-5 system card, 2025b. URL https://cdn.openai.com/gpt-5-system-card.pdf.

OpenAI. OpenAI o3 and o4-mini — system card, 2025c. URL https://openai.com/index/o3-o4-mini-system-card/.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract-Conference.html.

Christos H. Papadimitriou and John N. Tsitsiklis. The complexity of markov decision processes. *Mathematics of Operations Research*, 12(3):441–450, 1987. ISSN 0364765X, 15265471. URL http://www.jstor.org/stable/3689975.

Top Piriyakulkij, Cassidy Langenfeld, Tuan Anh Le, and Kevin Ellis. Doing experiments and revising rules with natural language and probabilistic reasoning. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/5f1b350fc0c2affd56f465faa36be343-Abstract-Conference.html.

Wasu Top Piriyakulkij, Volodymyr Kuleshov, and Kevin Ellis. Active Preference Inference using Language Models and Probabilistic Reasoning, 2023. URL https://arxiv.org/abs/2312.12009.

Gabriel Poesia, David Broman, Nick Haber, and Noah D. Goodman. Learning formal mathematics from intrinsic motivation. In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/4b8001fc75f0532827472ea5a16af9ca-Abstract-Conference.html.

Linlu Qiu, Fei Sha, Kelsey Allen, Yoon Kim, Tal Linzen, and Sjoerd van Steenkiste. Bayesian Teaching Enables Probabilistic Reasoning in Large Language Models. *ArXiv preprint*, abs/2503.17523, 2025. URL https://arxiv.org/abs/2503.17523.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

Tom Rainforth, Adam Foster, Desi R. Ivanova, and Freddie Bickford Smith. Modern Bayesian Experimental Design, 2023. URL https://arxiv.org/abs/2302.14545.

Sudha Rao and Hal Daumé III. Learning to ask good questions: Ranking clarification questions using neural expected value of perfect information. In Iryna Gurevych and Yusuke Miyao (eds.), *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2737–2746, Melbourne, Australia, 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1255. URL https://aclanthology.org/P18-1255.

Anselm Rothe, Brenden M. Lake, and Todd M. Gureckis. Question asking as program generation. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 1046–1055, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/24681928425f5a9133504de568f5f6df-Abstract.html.

Anselm Rothe, Brenden M. Lake, and Todd M. Gureckis. Do People Ask Good Questions? *Computational Brain & Behavior*, 1(1):69–89, 2018. ISSN 2522-0861, 2522-087X. doi: 10.1007/s42113-018-0005-5. URL http://link.springer.com/10.1007/s42113-018-0005-5.

Anselm Rothe, Brenden M. Lake, and Todd M. Gureckis. Asking goal-oriented questions and learning from answers. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 41(0), 2019. URL https://escholarship.org/uc/item/04q6z1mq.

Azzurra Ruggeri, Tania Lombrozo, Thomas L Griffiths, and Fei Xu. Sources of developmental change in the efficiency of information search. *Developmental psychology*, 52(12):2159, 2016. Publisher: American Psychological Association.

Samuel Schmidgall, Yusheng Su, Ze Wang, Ximeng Sun, Jialian Wu, Xiaodong Yu, Jiang Liu, Michael Moor, Zicheng Liu, and Emad Barsoum. Agent Laboratory: Using LLM Agents as Research Assistants, 2025. URL https://arxiv.org/abs/2501.04227.

C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. ISSN 0005-8580. doi: 10.1002/j.1538-7305.1948.tb01338.x. URL https://ieeexplore.ieee.org/document/6773024.

Dan Sperber, Fabrice Clément, Christophe Heintz, Olivier Mascaro, Hugo Mercier, Gloria Origgi, and Deirdre Wilson. Epistemic Vigilance. *Mind & Language*, 25(4):359–393, 2010. ISSN 1468-0017. doi: 10.1111/j.1468-0017.2010.01394.x. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1468-0017.2010.01394.x. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1468-0017.2010.01394.x.

Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: An introduction (2nd ed.)*. MIT Press, 2018. URL http://incompleteideas.net/book/the-book-2nd.html.

Edward Vul, Noah Goodman, Thomas L Griffiths, and Joshua B Tenenbaum. One and done? Optimal decisions from very few samples. *Cognitive science*, 38(4):599–637, 2014. Publisher: Wiley Online Library.

Ruocheng Wang, Eric Zelikman, Gabriel Poesia, Yewen Pu, Nick Haber, and Noah D. Goodman. Hypothesis search: Inductive reasoning with language models. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=G7UtIGQmjm.

Ruoyao Wang, Peter Jansen, Marc-Alexandre Côté, and Prithviraj Ammanabrolu. ScienceWorld: Is your agent smarter than a 5th grader? In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (eds.), *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 11279–11298, Abu Dhabi, United Arab Emirates, 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.emnlp-main.775. URL https://aclanthology.org/2022.emnlp-main.775.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/9d5609613524ecf4f15af0f7b31abca4-Abstract-Conference.html.

Lionel Wong, Gabriel Grand, Alexander K. Lew, Noah D. Goodman, Vikash K. Mansinghka, Jacob Andreas, and Joshua B. Tenenbaum. From Word Models to World Models: Translating from Natural Language to the Probabilistic Language of Thought, 2023. URL https://arxiv.org/abs/2306.12672.

Lionel Wong, Katherine M. Collins, Lance Ying, Cedegao E. Zhang, Adrian Weller, Tobias Gerstenberg, Timothy O'Donnell, Alexander Lew, Jacob Andreas, Tyler Brooke-Wilson, and Joshua B. Tenenbaum. Modeling Open-World Cognition as On-Demand Synthesis of Probabilistic Models. *Proceedings of the Annual Meeting of the Cognitive Science Society*, 47(0), 2025. URL https://escholarship.org/uc/item/4j38w2tg.

Jianzhu Yao, Kevin Wang, Ryan Hsieh, Haisu Zhou, Tianqing Zou, Zerui Cheng, Zhangyang Wang, and Pramod Viswanath. SPIN-Bench: How Well Do LLMs Plan Strategically and Reason Socially?, 2025. URL https://arxiv.org/abs/2503.12349.

Lance Ying, Tan Zhi-Xuan, Lionel Wong, Vikash Mansinghka, and Joshua Tenenbaum. Grounding Language about Belief in a Bayesian Theory-of-Mind. *ArXiv preprint*, abs/2402.10416, 2024. URL https://arxiv.org/abs/2402.10416.

Lance Ying, Ryan Truong, Katherine M. Collins, Cedegao E. Zhang, Megan Wei, Tyler Brooke-Wilson, Tan Zhi-Xuan, Lionel Wong, and Joshua B. Tenenbaum. Language-Informed Synthesis of Rational Agent Models for Grounded Theory-of-Mind Reasoning On-The-Fly, 2025. URL https://arxiv.org/abs/2506.16755.

Lili Yu, Howard Chen, Sida I. Wang, Tao Lei, and Yoav Artzi. Interactive classification by asking informative questions. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 2664–2680, Online, 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.237. URL https://aclanthology.org/2020.acl-main.237.

Michael J. Q. Zhang and Eunsol Choi. Clarify When Necessary: Resolving Ambiguity Through Interaction with LMs, 2023. URL https://arxiv.org/abs/2311.09469.

Appendix

Table of Contents

A Battleship

A.1 Extended Game Description and Prior Work

Our environment draws inspiration from the cognitive science literature, where Battleship-like tasks have previously been utilized to study human information-seeking behavior. In prior work, single-player participants viewed partially-revealed game boards and decided what tiles to reveal (Gureckis & Markant, 2009; Markant & Gureckis, 2012; 2014) or what questions to ask (Rothe et al., 2017; 2018; 2019). In particular, Rothe et al. developed a novel approach where individual questions for specific intermediate game states were elicited from participants and translated into programs in a domain-specific language (DSL) in order to compute various formal characteristics, such as the expected information gain (EIG), minimum description length (MDL), and answer type (Boolean, number, color, etc.). A log-linear model was fit to human data in order to determine relative feature importance and to score the likelihood of questions sampled from a generative grammar defined over the DSL.

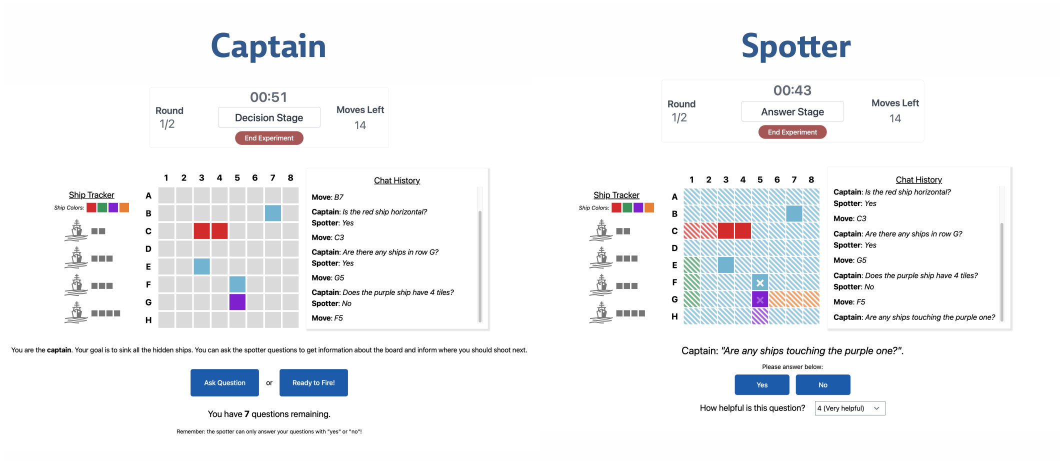
Collaborative Battleship In our *Collaborative Battleship* game (Fig. 1), players alternate between two roles: the Captain must efficiently navigate explore/exploit tradeoffs in order to gain information about the board through questions and actions, while the Spotter must provide grounded answers to (possibly ambiguous) questions given full visibility of the board. Games start from a blank board; at every turn, the Captain chooses whether to (a) reveal a tile by "shooting" or (b) ask a question. When a question is asked, the Spotter—who sees the full board—is prompted to answer it. Captains are given a limited budget of 15 questions and 40 shots, which under mild assumptions is sufficient to sink all ships. The game ends when either all ships are sunk (team win) or the Captain exhausts their shot budget (team loss). Further details about the game rules, interface, and data collection are provided in §A.2.

Key methods contributions Relative to prior studies, our work extends the Battleship paradigm in several key dimensions. (1) *Full multi-turn games:* Prior work was limited to one or two-turn snapshots; here, we simulate full games consisting of interleaved sequences of up to 40 moves and 15 questions. (2) *Two-player setting:* In the multi-turn setting, players need to receive real-time answers to their questions; to facilitate this interaction loop, we introduce a second player to act as a "Spotter," detailed above. (3) *Python programs:* In prior work, question-meanings were manually translated into programs. While follow-up work from Grand et al. (2024) proposed using LMs to automate this process, programs were still restricted to a hand-engineered DSL. Here, we represent questions with Python programs, which are both more expressive and easier in practice for LMs to generate. (4) *State space complexity:* Prior work used small 6×6 boards; we expand to 8×8 boards, which allow for more ships and position combinations and make question-asking more integral to skilled play.

Information Bottleneck There is also one key aspect in which our setting is intentionally more *restricted*. Prior work placed no formal restrictions on questions; in principle, this meant that players could ask questions like, "What are the coordinates of all the ships?"[2] Rather than attempt to detect such questions in real-time, we instead simply restrict the Spotter to answer with "Yes" or "No." As discussed in §3, this rule acts as an *information bottleneck* that limits the channel capacity. While less open-ended than the original setting, this approach is much more directly-enforceable, which is critical in our setting where human players are incentivized with real monetary rewards (§4.1), and where teams can and do form ad-hoc conventions across rounds.

[2]To discourage "game-breaking" questions, Rothe et al. (2017) instructed participants to only ask questions that could be answered with a single word. Nevertheless, there are many ways of constructing a nominally "one-word" answer that encodes the game state (e.g., *RedB1B3PurpF4F7...*). Since the language of such encodings is undecidable, we instead opt to enforce a binary information bottleneck.

A.2 Human Study Details



Move: *B7*
Captain: *Is the red ship horizontal?*
Spotter: *Yes*
Move: *C3*
Captain: *Are there any ships in row G?*
Spotter: *Yes*
Move: *G5*
Captain: *Does the purple ship have 4 tiles?*
Spotter: *No*
Move: *F5*

You are the **captain**. Your goal is to sink all the hidden ships. You can ask the spotter questions to get information about the board and inform where you should shoot next.

You have **7** questions remaining.

Remember: the spotter can only answer your questions with "yes" or "no"!

Captain: *Is the red ship horizontal?*
Spotter: *Yes*
Move: *C3*
Captain: *Are there any ships in row G?*
Spotter: *Yes*
Move: *G5*
Captain: *Does the purple ship have 4 tiles?*
Spotter: *No*
Move: *F5*
Captain: *Are any ships touching the purple one?*

Captain: *"Are any ships touching the purple one?"*.

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).

A.2.1 Study Setup

We recruited N=42 participants from Prolific (www.prolific.com). Participants were required to be located in the United States, fluent in English, and maintain a Prolific approval rate of at least 95%.

The study was conducted with IRB approval in multiple sessions that occurred between November 26, 2024 and January 31, 2025. We conducted the study in a synchronous, two-player format using the Empirica platform (Almaatouq et al., 2021). Participants were matched via Empirica's built-in matchmaking system, which pairs participants as they arrive. Participants were required to use a desktop or laptop computer with a modern web browser (Chrome, Firefox, Edge, or Safari). Mobile devices were not supported.

The human study included a tutorial to familiarize participants with the rules of the game; we did not assume prior experience with Battleship. However, we did set specific criteria on Prolific to exclude participants who had previously completed any prior study (e.g., pilot studies) from our group that specifically used the Collaborative Battleship protocol.

Median time to complete the study was 48–60 minutes, depending on the session. Participants earned a base rate of \$8.00 for completing the study, with performance-contingent bonuses of \$0.20 per hit and -\$0.10 per miss (with a bonus floor of \$0 such that base pay was always guaranteed). In total, average per-participant earnings ranged from \$12.03–\$14.62 / hr.

A.2.2 Stimulus and Block Design

We sampled $v = 18$ unique Battleship boards from the full design space as experimental stimuli. Each board was an 8×8 grid containing exactly four ships—with possible lengths $2, 3, 4,$ and 5—colored red, purple, green, and orange. Boards were labeled B01–B18.

Participants were allocated to boards via an equal-replication incomplete block design. With $N = 42$ participants organized into 21 pairs (blocks; $b = 21$), each pair was randomly assigned $k = 6$ boards. Within each pair, players alternated Captain and Spotter roles across rounds, and board order was randomized. This allocation was designed to ensure uniform coverage: each board appeared in exactly $r = 7$ pairs, satisfying $vr = bk$ $(18 \times 7 = 21 \times 6)$.

A.2.3 Participant Instructions

Below, we reproduce the full instructions provided to participants before starting the study.

Please read all instructions before accepting this study.

In this study, you will play a collaborative version of the board game **Battleship** with another Prolific participant.

- As Captain, you will ask your partner questions about the board to try to get information about the hidden ships.
- As Spotter, you will answer questions for your partner, given special knowledge of the ship locations.

Your goal is to reveal all the ships in **as few moves as possible**. By working together effectively as a team, you and your partner have the opportunity to **earn double** though bonus payment!

Earning bonus payment

In addition to the hourly rate, you and your partner are each eligible for a **bonus of up to $12.00** ($2.00 per round). Your bonus will be determined by how well you work together as a team: the fewer moves it takes to sink all the ships, the higher your total bonus will be.

To maximize your team's bonus, the Captain will need to think carefully about what questions will be most informative, and the Spotter will need to answer the Captain's questions correctly. **Don't rush to complete the study; every hit increases your bonus, but every miss reduces your bonus, so take your time to think through your actions.**

As question-asking is a key part of this study's motivation, your bonus is contingent on asking thoughtful questions. We will not issue bonuses for poor quality / copy-pasted questions, move spamming, and other behaviors indicative of low-effort participation.

Your responsibilities as a participant

This study is designed to take approximately 1 hr. Past participants report the game to be fun and engaging. Nevertheless, this is also a cognitively-demanding task that will require your full, undivided attention. We ask that you only accept this study if you are prepared to spend the next hour working collaboratively with your partner to play through the game.

Stay engaged and be patient: Some portions of the game will require you to wait for a response from your partner. While participating in the study, we ask that you do not navigate away from the study, open other browser tabs, or otherwise attempt to multi-task. Please treat your partner as you would like to be treated by responding in a timely manner and not making them wait for extended periods.

Attention checks: To ensure that participants remain responsive throughout the whole study, **the game enforces a time limit on each move**. If you exceed this time limit (attention check #1), you will be shown a prompt asking to confirm whether you are still online (attention check #2). If you fail both of these attention checks, the study will end, and your submission will be rejected. (If your partner fails the attention checks, your submission will still be accepted, provided that you yourself participated in good faith.)

Manually terminating the study early: If at any point you encounter an issue that prevents you from completing the study, please press the "End Experiment" button at the top of the screen. You are still eligible for compensation. On the feedback screen, please make sure to copy the completion code and provide an explanation of the issue that prompted you to end the experiment. You may also message us.

Important details

Player ID: Please enter your **Prolific ID** as your player identifier on the intro screen. (Do not include @*email.prolific.com* in your Player ID.)

Completion code: Please make sure to copy the completion code we provide to you at the end of the study. This is necessary to ensure you receive payment.

Matchmaking: You may experience some initial wait times while we attempt to match you with another participant. If you are not matched within 10 minutes of starting the study, your game will end automatically, and you are eligible for compensation at the hourly rate (totaling \$2.50). In order to receive compensation for a matchmaking timeout, you **must** still return the study. If you do not return the study, we will not be able to issue payment.

Network latency: You may experience some screen flickering, which is due to network latency. If the screen goes blank for an extended period, you may refresh your browser. (Refreshing will not affect your progress in the study.) To minimize your chance of experiencing latency issues, please ensure that you are on a stable, fast internet connection.

Bug reports and feedback: Please report any bugs you encounter. Additional bonuses will be given for helpful bug reports and feedback.

A.3 BattleshipQA Dataset

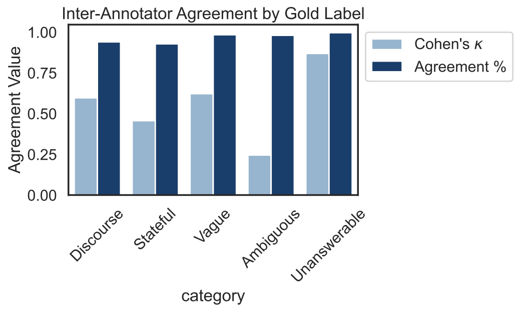
We manually annotated the human data in order to (1) establish a set of gold labels for evaluating model performance and (2) analyze the types of questions being asked. To validate the accuracy of human answers, we annotated all questions with a "gold answer" label; questions for which there is 100% inter-annotator agreement were retained. After this process, we obtained a gold dataset of 931 questions, establishing a human accuracy baseline of 92.5%. Additionally, we performed a detailed analysis of the question types present in the dataset according to the following taxonomy:

- **Simple:** Simple questions require only the current board state x_t to answer. For analysis purposes, we designate any question with one or more of the following labels as "complex."
- **Stateful:** Stateful questions require prior board state history $\{x_1, \dots, x_{t-1}\}$ (e.g., "Are any ships touching where I just fired?").
- **Discourse:** Discourse questions require prior conversation history between players $\{(q_1, A_1), \dots, (q_{t-1}, A_{t-1})\}$ (e.g., due to ad-hoc convention formation).
- **Vague:** Vague questions require resolving pragmatic aspects of question meaning (e.g., "middle," "edge," "near"). Prototypically, a scalar value that is unclear.
- **Ambiguous:** Ambiguous questions admit multiple possible interpretations, each of which could be answered in principle. (e.g., "Is there a ship on F after 5 or on G?" could refer to either part or all of row G.)
- **Unanswerable:** Unanswerable questions are not interpretable even with full context and history. Typically, these do not admit a yes/no answer (e.g., "Where is the green ship?"), or are based on a false premise/misunderstanding about the board state. These questions were removed from the dataset.

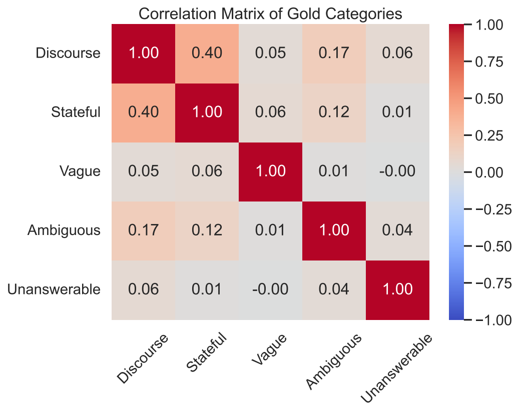
Table 2 shows the distribution of labels in the dataset. Inter-annotator agreement ranged from 94.0–99.6%; further details are given in Fig. 7a.

Gold Label	Count	Percent
Simple	546	58.6
Discourse	212	22.8
Stateful	205	22.0
Vague	49	5.3
Ambiguous	37	4.0
Overall	931	100.0

Table 2: Expert-labeled attributes of human questions. (Questions are multi-label; definitions in §A.3.)



(a) Inter-annotator agreement statistics across question labels.



	Discourse	Stateful	Vague	Ambiguous	Unanswerable
Discourse	1.00	0.40	0.05	0.17	0.06
Stateful	0.40	1.00	0.06	0.12	0.01
Vague	0.05	0.06	1.00	0.01	-0.00
Ambiguous	0.17	0.12	0.01	1.00	0.04
Unanswerable	0.06	0.01	-0.00	0.04	1.00

(b) Pairwise Pearson correlations between annotation dimensions.

Figure 7: **Annotation quality analyses.** (a) Inter-annotator agreement demonstrating reliability of gold labels. (b) Correlation structure among annotated dimensions, indicating which aspects of questions co-vary.

A.4 Illustrated Examples

We show selected examples to illustrate the question categories defined in §A.3 as well as other behavioral phenomena of interest in BattleshipQA. Spotter-generated code translations for these examples are provided in §D.1.1. While these examples are handpicked, we also provide a set of *randomly-sampled* question contexts for both humans and models in Tables 6 and 7.

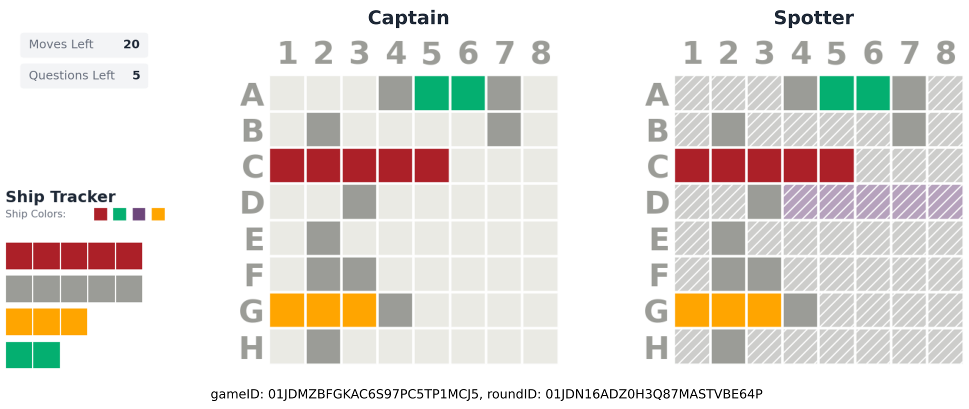


Figure 8: **Simple question.** In this example, the Captain asks, “Does E7 have a part of a ship?” and the Spotter responds, “No.” This question is a textbook example of a simple question, as it can be answered using only the true board state without any additional context.

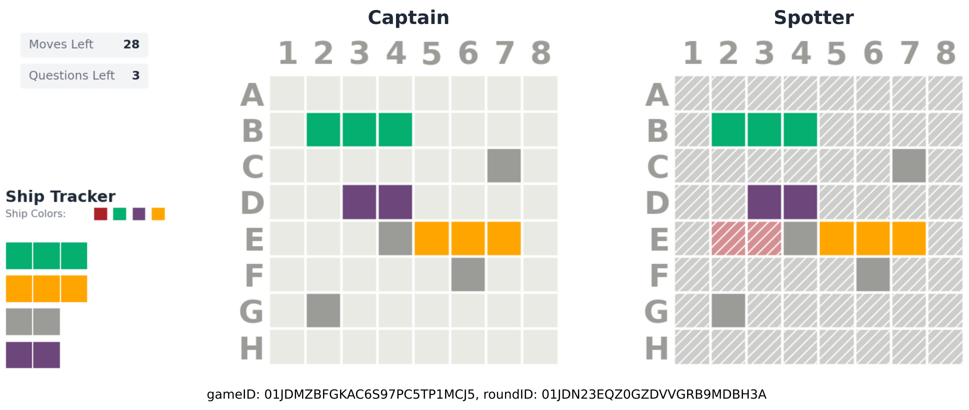


Figure 9: **Stateful question.** In this example, the Captain asks, “any parts of an undiscovered boat in Row e?”, to which the Spotter responds “Yes.” This question is a textbook example of a stateful question, as it requires the Spotter to reason about what is currently visible in the Captain view.

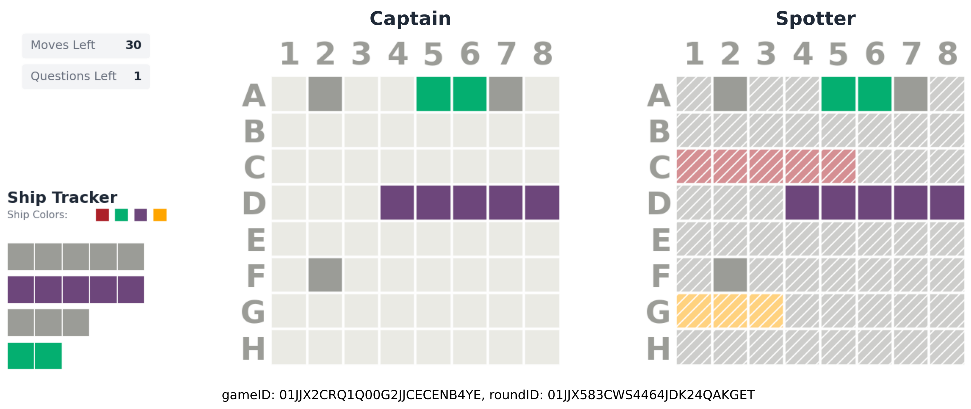


Figure 10: **Discourse-dependent questions.** In this example, the Captain asks a series of three questions in a row: “Is there any horizontal ships left?” (sic); “is it in either row c or e?”; “Does it touch E4 or C4?” The first question establishes a discourse context (remaining horizontal ships), making the subsequent questions discourse-dependent. Without this context, the second and third questions are ambiguous (e.g., “it” could refer to any ship).

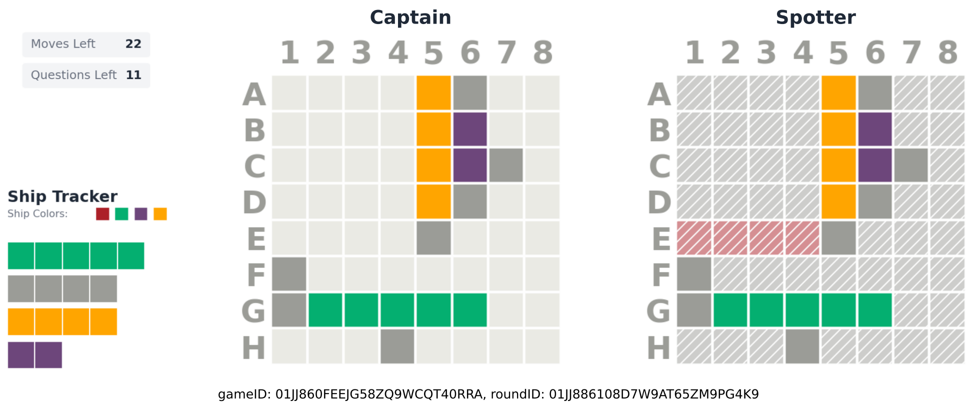


Figure 11: **Vague question.** In this example, the Captain asks, “Is red close to green?” This is a textbook example of a vague question as it requires pragmatic interpretation of a fuzzy concept of “closeness” that is inherently subjective. Case in point: while the human Spotter answered, “No” here, the expert annotators unanimously labeled the answer as “Yes.”

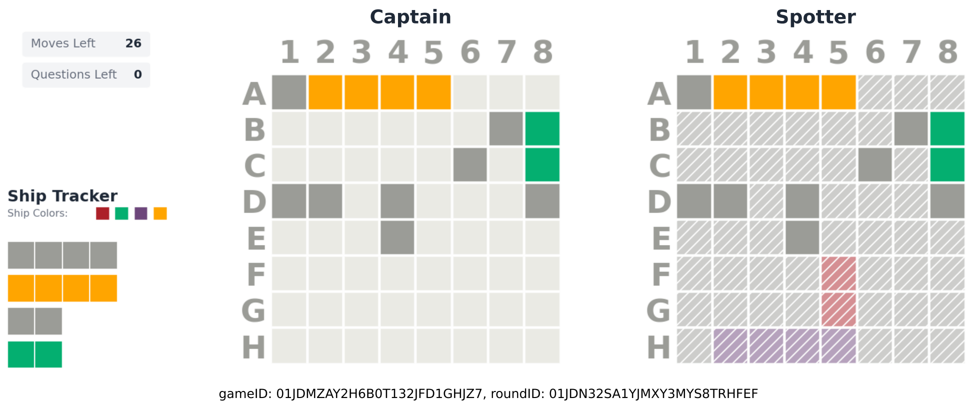


Figure 12: **Ambiguous question.** In this example, the Captain asks, “is one on f after 5 or on g?” This question has multiple possible logical interpretations. It seems fairly clear that the Captain wants to know whether there are any ships on F6-F8. However, in terms of Row G, it is ambiguous whether the Captain is interested in *the corresponding portion* (G6-G8), or the *whole row*. In contrast to the vague questions (e.g., Fig. 11), both interepretations are well-defined; however, the ambiguity forces the Spotter to choose between multiple possible interpretations when considering an answer.

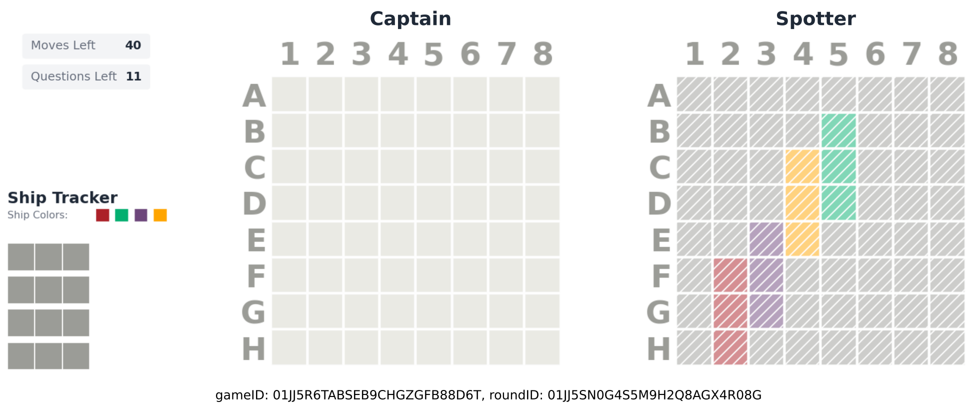


Figure 13: **Unanswerable question.** In this example, the Captain asks, "where are the ships?" which cannot be answered with "Yes" / "No." Moreover, it is a good motivating example for the need to design *Collaborative Battleship* with a binary information bottleneck in the first place, as discussed in §2.

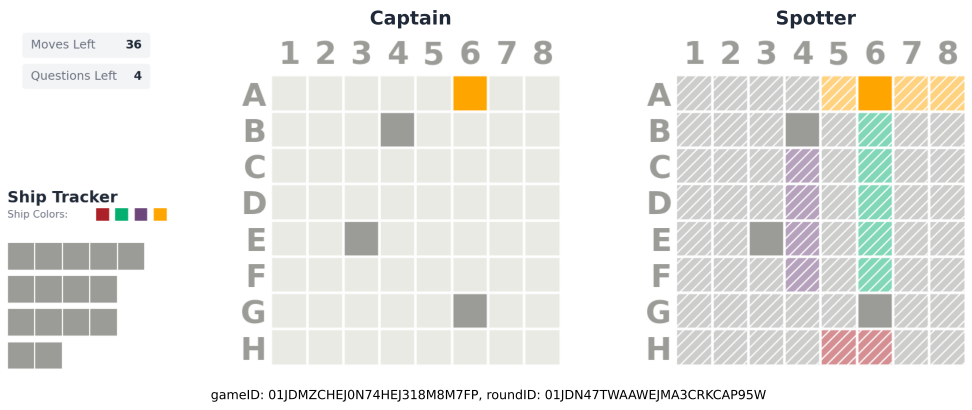


Figure 14: **Benevolent lying.** In this example, the Captain asks, "Is orange verticle?" (sic), to which the Spotter responds "Yes." Even though orange is not vertical, the Spotter's answer is distinctly *helpful* in context, as it causes the Captain to fire at B6 on the subsequent turn, revealing a tile of the Green ship.

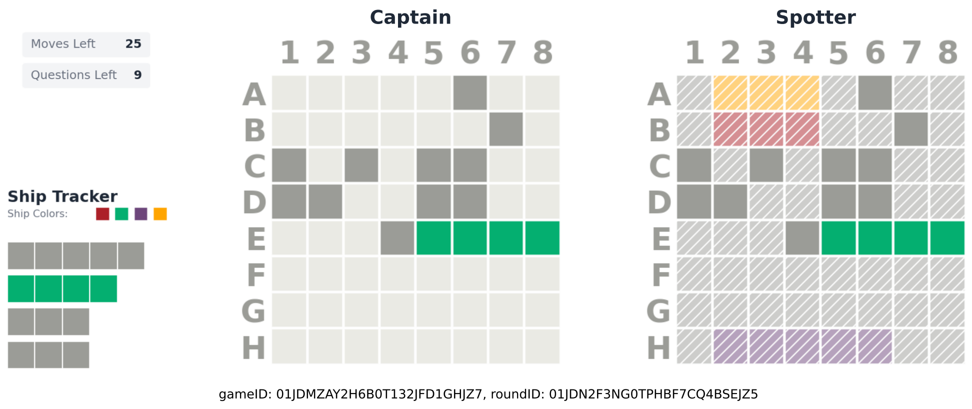


Figure 15: **Epistemic vigilance.** In this fascinating exchange, which plays out over multiple games, the Captain starts by asking whether there is a ship "on or before c6 or d6?" to which the Spotter responds, "Yes." This prompts the Captain to fire successively at multiple tiles in this region, resulting in a prolonged miss streak. Frustrated, the Captain embeds an accusation in their question: "you lied.... Is it on row b?" (to which the Spotter responds "Yes"). On the *subsequent game*, the Spotter, now acting as the Captain, writes "Nobody lied," clarifying that in their view, "A2 IS on or before c6 or d6." This exchange highlights the behavioral phenomenon of *epistemic vigilance* (Sperber et al., 2010), where people are sensitive to the reliability of information provided by others, and will actively monitor for potential deception or misunderstanding.

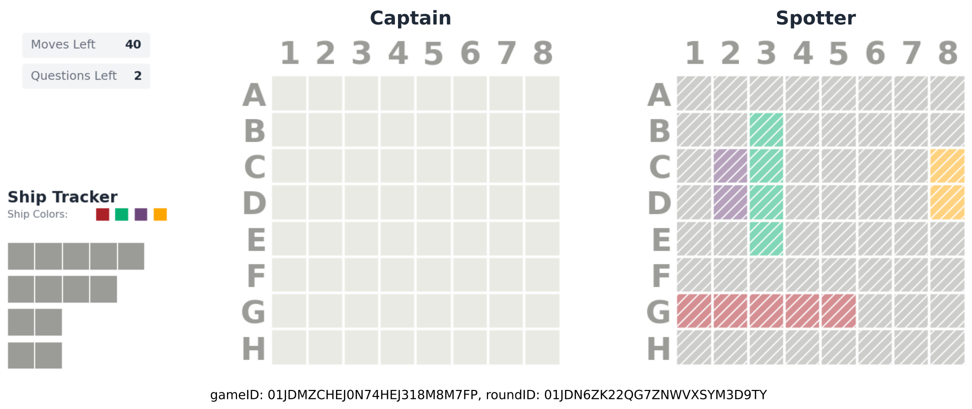


Figure 16: **Phatic communication.** In this example, the Captain starts the round by exclaiming, "We killin it bro lets do this! LETS DO THIS! lol," before getting down to business: "Alright A - C , 1 - 4?" This example illustrates some colorful and undeniably human behaviors that were elicited by our task. Specifically, a surprising number of participants embedded "side comments" into their questions—a clever way to interact with their partner given the constraints of the interface. This behavior is reminiscent of *phatic communication* (Malinowski, 1927), where the primary function of an utterance is social rather than informational. In this case, the Captain's pep talk serves to build rapport and boost morale, which may be particularly important in a collaborative task with real monetary incentives on the line.

B Sequential Monte Carlo Approximation

All of the modeling strategies that we consider involve posterior inference over possible world states $s \in \mathcal{S}_{\vdash x}$ given the current observation history $\mathcal{H}_{1:t} = \{(q_\tau, \tilde{a}_\tau)\}_{\tau=1}^{t}$. However, the space of possible Battleship boards is extremely large ($\approx 2^{30}$ valid configurations). While it is technically feasible to infer $\pi_t(s) = p(s \mid x, \mathcal{H}_{1:t})$ exactly via enumeration, this approach is intractible for other key quantities of interest (e.g., computing p_t and EIG, which require function execution).

We therefore use a sequential Monte Carlo (SMC; Doucet et al., 2001) approach to maintain a particle approximation to π_t.

Particles. A weighted set $\{(s_j, w_j^{(t)})\}_{j=1}^{N}$ with $\sum_j w_j^{(t)} = 1$, where each $s_j \in \mathcal{S}_{\vdash x}$ is a valid Battleship board configuration, and $w_j^{(t)}$ is the weight of particle j at time t. We initialize particles by sampling $s_j \sim p(s)$ from the prior distribution over boards, and setting $w_j^{(0)} = 1/N$.

Posterior update given an observation. After observing $\tilde{a}_t$, reweight and renormalize:

$$\tilde{w}_j \leftarrow w_j^{(t)} \Big[(1-\varepsilon)\,\mathbf{1}\{\tilde{a}_t = f_{q_t}(s_j)\} + \varepsilon\,\mathbf{1}\{\tilde{a}_t \neq f_{q_t}(s_j)\}\Big], \quad w_j^{(t+1)} \leftarrow \tilde{w}_j \Big/ \sum_{k=1}^{N} \tilde{w}_k. \tag{8}$$

Scoring a candidate q**.** Estimate p_t and EIG by

$$\widehat{p}_t(q) = \sum_{j=1}^{N} w_j^{(t)}\,\mathbf{1}\{f_q(s_j) = 1\}\,, \qquad \widehat{\mathrm{EIG}}_\varepsilon(q) = \mathrm{H}_b\big(\varepsilon + (1-2\varepsilon)\,\widehat{p}_t(q)\big) - \mathrm{H}_b(\varepsilon). \tag{9}$$

Intuition. $\widehat{p}_t(q)$ is just the fraction of our current probability mass that predicts a “yes” to this question.

C Prompt Library

C.1 Shared Components

C.1.1 Game Setup Prompt

System

```
You are playing the board game Battleship. In this variant of the game, pairs of
players collaborate as a team to find the location of ships on the board.
Each player is assigned to one of two roles: the 'Captain' or the 'Spotter'.
The Captain's role is to decide when and where to reveal tiles on the board. On
each turn, the Captain can ask the Spotter a question about the board, or make a
move by guessing a tile that they think contains a ship.
The Spotter's role is to provide the Captain with information about the hidden
tiles. The Spotter has full visibility of the board, but can only answer the
Captain's questions with 'Yes' or 'No'.

The board is an 8x8 grid, with lettered rows A, B, C, D, E, F, G, H and numbered
columns 1, 2, 3, 4, 5, 6, 7, 8.
Coordinates are specified as a row, column pair. For example, C2 is the tile in row
C, column 2.
There are four ships on the board: Green, Red, Purple, and Orange.
Ships are oriented either horizontally or vertically and range from 2 to 5 tiles in
length.

The board is represented as a numpy array with the following symbols:
-1: Hidden
0: Water
1: Red ship
2: Green ship
3: Purple ship
4: Orange ship

{{board}}

The ships on the board are of the following lengths: {{lengths}}. {{ship_tracker
for ship in ships}}
```

Ship Tracker: Unsunk Ship

```
A ship of length {{length}} is not yet sunk.
```

Ship Tracker: Sunk Ship

```
A ship of length {{length}} has been sunk. It was the {{ship_color_name}}.
```

C.1.2 Reasoning Options

Reasoning: Direct

```
Return your answer directly. Do not include any extra reasoning or explanation.
```

Reasoning: Chain-of-Thought

```
Please think step-by-step about the task before returning your answer.
```

C.2 Captain Prompt

C.2.1 Task Prompts

Task: Decision

```
You will be given a partially-revealed game board.
Your task is to choose whether you'd like to ask a question about the board to gain
more information, or make a move by guessing a tile that you think contains a ship.
Please answer in a single word: 'Question' or 'Move', and enclose your final answer
in <answer></answer> tags, e.g. <answer>Question</answer> or <answer>Move</answer>.
```

Task: Move

```
You will be given a partially-revealed game board.
Your task is to give the coordinates of the hidden tile you think is most likely to
contain a ship tile.
Hidden tiles are marked by '-1'.
Respond with only the coordinates (e.g., A1, B2, etc.), and enclose your answer in
<answer></answer> tags, e.g. <answer>A1</answer>.
```

Task: Question

```
You will be given a partially-revealed game board.
Your task is to ask a single question that will help you gain the most information
possible about the position of the remaining hidden ships on the board.
You can ask any question, but it must be answerable with a Boolean answer (Yes/No).
Make sure to enclose your question in <answer></answer> tags, e.g. <answer>Is the
sky blue?</answer>.
```

C.2.2 Full Template

Captain

```
{{Game Setup Prompt}}

Here is the current board:
{{current_board_numpy_array}}

You are playing as the Captain. Your objective is to find all the ships on the
board as efficiently as possible.

<Insert one of the task prompts above (Decision / Move / Question) here>

You can ask {{questions_remaining}} more questions over the course of the game, and
can fire {{moves_remaining}} more times.
Ship Status: {{sunk_status}}

Please think step-by-step about the task before returning your answer.
```

C.3 Spotter Prompt

C.3.1 Variants

Spotter: Direct

```
Remember: You can only answer with 'Yes' or 'No'. Please only answer with a single
word. Enclose your answer in <answer></answer> tags, e.g. <answer>Yes</answer> or
<answer>No</answer>.
```

Spotter: Code

~~~
Your task is to write a Python function that computes the answer to the question.

The function should accept two numpy arrays as arguments: `true_board` and
`partial_board`.
The `true_board` is the full board, which is only visible to you as the Spotter.
The `partial_board` is the current board that is visible to the captain, which may
contain hidden tiles. Your function should return a Boolean value, which will be
interpreted as 'Yes' or 'No'.

Your function should be defined generically to work with any true and partial
board, not just the ones you are given. This means that your function must perform
some operations on `true_board`, `partial_board`, or both boards in order to
compute the answer to the Captain's question. Avoid hardcoding the answer.
In some situations, the correct answer may depend on the current state of the game.
For instance, if the Captain asks, 'Are there any ships in Row A?', the answer
depends on what ships have already been revealed. If there are any unrevealed ship
tiles in Row A, then the answer is 'Yes'. However, if all ships in Row A have
already been revealed, then the correct answer is 'No'. Comparing the
`partial_board` with the `true_board` will allow you to determine which ship tiles
remain unrevealed. Remember: Your goal is to help the Captain find the location of
the ships on the board, so your function should be designed to provide the useful
information in context.

Your function should be defined as follows:
```python
def answer(true_board: np.ndarray, partial_board: np.ndarray) -> bool:
    # Your code here
    return ANSWER
```

Your code will be executed in an environment with `numpy` (namespaced as `np`) and
a `board` variable (a numpy representation of the board).
Make sure your code is valid Python and does not contain any syntax errors.
You are responsible for implementing the `answer()` function, but do not invoke it
or include any other code.
~~~

C.3.2 Full Template

Spotter

```
{{Game Setup Prompt}}

Here is the partial board, which is the view that is visible to the Captain:
{{partial_board_numpy_array}}

Here is the full board, which only you as Spotter have access to:
```

```
{{true_board_numpy_array}}

You are playing as the Spotter. Your objective is to answer the Captain's questions
as accurately as possible.

<Insert one of the Spotter variants above (Direct / Code) here>

<Insert one of the reasoning options above (Direct / Chain-of-Thought) here>

Here is the question the Captain asked:
Captain (question): {{question_text}}
```

D EVALUATIONS

D.1 SPOTTERQA EVALUATIONS

MODEL	SPOTTER	OVERALL	SIMPLE	COMPLEX	DISCOURSE	STATEFUL	VAGUE	AMBIG.	VALID
Human									
–	–	0.925	0.928	0.919	0.962	0.941	0.816	0.865	1.000
Anthropic									
claude-opus-4	Base	0.868	0.893	0.833	0.836	0.820	0.843	0.816	0.999
	CoT	0.906	0.936	0.864	0.869	0.849	0.843	0.816	0.999
	Code	0.937	0.970	0.889	0.859	0.878	0.922	0.868	0.993
	CoT + Code	0.944	0.977	0.897	0.892	0.883	0.922	0.789	0.998
claude-sonnet-4	Base	0.834	0.869	0.784	0.765	0.795	0.725	0.868	1.000
	CoT	0.904	0.941	0.851	0.831	0.810	0.941	0.816	0.999
	Code	0.939	0.966	0.900	0.906	0.888	0.882	0.816	0.996
	CoT + Code	0.937	0.970	0.889	0.883	0.888	0.902	0.816	0.997
Google									
gemini-2.5-flash	Base	0.231	0.293	0.141	0.080	0.117	0.294	0.237	0.306
	CoT	0.516	0.564	0.447	0.385	0.424	0.588	0.579	0.578
	Code	0.440	0.478	0.386	0.300	0.439	0.490	0.447	0.470
	CoT + Code	0.497	0.542	0.432	0.362	0.439	0.588	0.553	0.525
Meta Llama									
llama-3.1-405b-instruct	Base	0.701	0.771	0.602	0.568	0.600	0.608	0.632	0.998
	CoT	0.711	0.760	0.640	0.624	0.659	0.647	0.632	1.000
	Code	0.814	0.900	0.692	0.681	0.688	0.686	0.658	0.981
	CoT + Code	0.823	0.907	0.702	0.638	0.707	0.765	0.711	0.985
llama-3.1-70b-instruct	Base	0.622	0.676	0.545	0.512	0.556	0.549	0.605	1.000
	CoT	0.608	0.660	0.532	0.526	0.517	0.431	0.658	1.000
	Code	0.778	0.852	0.674	0.638	0.673	0.725	0.684	0.973
	CoT + Code	0.795	0.884	0.668	0.624	0.663	0.686	0.605	0.974
llama-4-maverick	Base	0.690	0.742	0.614	0.606	0.605	0.667	0.658	1.000
	CoT	0.789	0.864	0.681	0.648	0.644	0.686	0.789	1.000
	Code	0.797	0.862	0.704	0.662	0.741	0.686	0.632	0.977
	CoT + Code	0.845	0.911	0.751	0.742	0.741	0.784	0.632	0.976
llama-4-scout	Base	0.622	0.680	0.540	0.507	0.512	0.549	0.737	0.997
	CoT	0.690	0.757	0.594	0.577	0.580	0.647	0.605	0.999
	Code	0.767	0.839	0.663	0.601	0.634	0.745	0.711	0.958
	CoT + Code	0.784	0.864	0.668	0.620	0.663	0.804	0.605	0.958
OpenAI									
gpt-4.1	Base	0.752	0.805	0.676	0.700	0.673	0.647	0.579	1.000
	CoT	0.750	0.798	0.681	0.700	0.678	0.627	0.605	1.000
	Code	0.904	0.961	0.823	0.808	0.815	0.843	0.711	0.997
	CoT + Code	0.909	0.959	0.838	0.817	0.829	0.843	0.763	0.999
gpt-4.1-mini	Base	0.686	0.741	0.607	0.563	0.580	0.765	0.553	1.000
	CoT	0.783	0.857	0.676	0.620	0.668	0.824	0.658	1.000
	Code	0.899	0.957	0.815	0.793	0.810	0.902	0.763	0.998
	CoT + Code	0.896	0.959	0.805	0.789	0.780	0.902	0.711	0.999
gpt-4.1-nano	Base	0.589	0.606	0.563	0.516	0.580	0.608	0.684	1.000
	CoT	0.690	0.750	0.604	0.554	0.571	0.745	0.632	1.000
	Code	0.808	0.887	0.694	0.657	0.702	0.706	0.711	0.982
	CoT + Code	0.851	0.923	0.748	0.737	0.751	0.843	0.632	0.987
gpt-4o	Base	0.677	0.728	0.604	0.592	0.590	0.627	0.421	0.988
	CoT	0.808	0.862	0.730	0.723	0.717	0.725	0.711	1.000
	Code	0.878	0.957	0.763	0.709	0.751	0.843	0.737	0.995
	CoT + Code	0.888	0.953	0.794	0.770	0.800	0.804	0.737	0.995
gpt-4o-mini	Base	0.525	0.528	0.522	0.507	0.512	0.451	0.553	1.000
	CoT	0.591	0.630	0.535	0.531	0.488	0.471	0.579	1.000
	Code	0.799	0.862	0.707	0.685	0.654	0.922	0.658	0.992
	CoT + Code	0.828	0.905	0.717	0.718	0.702	0.804	0.658	0.982
gpt-5	Base	0.924	0.970	0.859	0.869	0.859	0.863	0.737	1.000
	CoT	0.926	0.964	0.871	0.873	0.863	0.882	0.737	1.000
	Code	0.939	0.980	0.879	0.878	0.888	0.863	0.789	0.997
	CoT + Code	0.940	0.980	0.882	0.878	0.883	0.902	0.737	0.999
o3	Base	0.928	0.966	0.874	0.864	0.863	0.882	0.816	1.000
	CoT	0.924	0.964	0.866	0.873	0.868	0.882	0.763	1.000
	Code	0.925	0.966	0.866	0.873	0.888	0.863	0.763	0.982
	CoT + Code	0.914	0.953	0.856	0.840	0.868	0.843	0.816	0.979
o4-mini	Base	0.927	0.953	0.889	0.887	0.873	0.922	0.789	1.000
	CoT	0.925	0.966	0.866	0.869	0.849	0.843	0.763	1.000
	Code	0.930	0.977	0.864	0.845	0.854	0.863	0.789	1.000
	CoT + Code	0.930	0.968	0.877	0.845	0.878	0.922	0.763	0.996

Table 3: SpotterQA: Accuracy breakdown by question type. *Valid* shows the proportion of answers returned by each model that followed formatting instructions (§C.3). With the exception of Gemini-2.5-Flash, which frequently returned invalid answers, all other models adhered readily to the answer format.

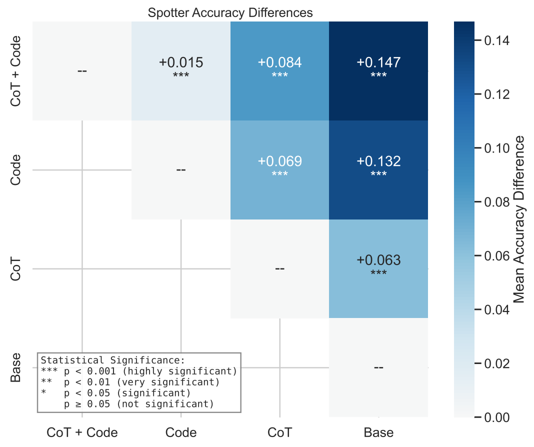


Figure 17: Differences in Spotter accuracy between conditions. Asterisks indicate statistical significance based on a two-sided paired t-test (* $p < 0.05$, ** $p < 0.01$, *** $p < 0.001$).

D.1.1 Example Spotter Programs

Program for Fig. 8, Simple Question, "Does E7 have a part of a ship?"

```
def answer(true_board: np.ndarray, partial_board: np.ndarray) -> bool:
    # Map the coordinate E7 to zero-based indices
    row_letter = 'E'
    col_number = 7
    r = ord(row_letter) - ord('A')  # 0-based row index
    c = col_number - 1              # 0-based column index

    # A ship tile is any non-zero value in the true board (1-4)
    return true_board[r, c] != 0
```

Program for Fig. 9, Stateful Question, "any parts of an undiscovered boat in Row e?"

```
def answer(true_board: np.ndarray, partial_board: np.ndarray) -> bool:
    # Column 5 in 1-based indexing corresponds to index 4 in 0-based indexing
    col_idx = 4
    # Safety check: if the boards don't have a column 5, return False
    if true_board.shape[1] <= col_idx or partial_board.shape[1] <= col_idx:
        return False

    true_col = true_board[:, col_idx]
    partial_col = partial_board[:, col_idx]

    # A "yet to be discovered" ship part is a cell that:
    # - contains a ship in the true board (>0)
    # - is not already revealed as that same ship value in the partial board
    undiscovered = (true_col > 0) & (partial_col != true_col)

    return bool(np.any(undiscovered))
```

Program for Fig. 10, Discourse-Dependent Question, "is it in either row c or e?"

```
import numpy as np

def answer(true_board: np.ndarray, partial_board: np.ndarray) -> bool:
    # Identify ship IDs present on the true board (positive numbers)
    ship_ids = [sid for sid in np.unique(true_board) if sid > 0]

    for sid in ship_ids:
        coords = np.argwhere(true_board == sid)
        if coords.size == 0:
            continue

        rows = coords[:, 0]
        cols = coords[:, 1]

        # Determine orientation: horizontal if all in the same row
        is_horizontal = np.unique(rows).size == 1

        if is_horizontal:
            # A ship is fully revealed only if all its tiles are revealed as its ID
on the partial board
            revealed_mask = (partial_board[rows, cols] == sid)
            fully_revealed = np.all(revealed_mask)

            # If any horizontal ship is not fully revealed, there are horizontal
ships left to find
            if not fully_revealed:
                return True

    # No horizontal ships remain unrevealed
    return False
```

Program for Fig. 11, Vague Question, “Is red close to green?”

```
def answer(true_board: np.ndarray, partial_board: np.ndarray) -> bool:
    # Define masks for red (1) and green (2)
    red = (true_board == 1)
    green = (true_board == 2)

    # Compute 4-neighbor adjacency: cells directly up/down/left/right of any red
cell
    neighbors_of_red = np.zeros_like(red, dtype=bool)
    # Down neighbors
    neighbors_of_red[1:, :] |= red[:-1, :]
    # Up neighbors
    neighbors_of_red[:-1, :] |= red[1:, :]
    # Right neighbors
    neighbors_of_red[:, 1:] |= red[:, :-1]
    # Left neighbors
    neighbors_of_red[:, :-1] |= red[:, 1:]

    # If any green cell is a 4-neighbor of a red cell, they are "close"
    return bool(np.any(neighbors_of_red & green))
```

Program for Fig. 12, Ambiguous Question, “is one on f after 5 or on g?”

```
def answer(true_board: np.ndarray, partial_board: np.ndarray) -> bool:
    # We interpret the question as:
    # "Is there any unrevealed ship tile either (a) on row F after column 5, or (b)
anywhere on row G?"
    # Rows are A-H => indices 0-7; thus F -> 5, G -> 6
    # "after 5" means columns 6-8 => 0-based column indices >= 5
    # Mask of unrevealed ship tiles (hidden in partial but ship in true)
    hidden_ship = (partial_board == -1) & (true_board > 0)

    # Create row/column index grids
    rows, cols = np.indices(true_board.shape)

    # Condition (a): on row F (index 5) with columns strictly after 5 (indices >= 5)
    cond_f_after5 = (rows == 5) & (cols >= 5)

    # Condition (b): on row G (index 6), any column
    cond_g_any = (rows == 6)

    # Combined area of interest
    area_mask = cond_f_after5 | cond_g_any

    # Check if any unrevealed ship tile lies in the area of interest
    return bool(np.any(hidden_ship & area_mask))
```

Program for Fig. 14, Benevolent Lying, “Is orange verticle?”

```
def answer(true_board: np.ndarray, partial_board: np.ndarray) -> bool:
    # Determine if the Orange ship (value 4) is oriented vertically on the true
board.
    coords = np.argwhere(true_board == 4)
    if coords.size == 0:
        # No orange tiles found; cannot be vertical
        return False
    rows = coords[:, 0]
    cols = coords[:, 1]
    unique_rows = np.unique(rows).size
    unique_cols = np.unique(cols).size
    # Vertical if all tiles share the same column and span multiple rows
    return (unique_cols == 1) and (unique_rows > 1)
```

Program for Fig. 15, Epistemic vigilance, “on or before c6 or d6?”

```
def answer(true_board: np.ndarray, partial_board: np.ndarray) -> bool:
    # Interpret the question as:
    # "Is there any unrevealed ship tile on or before C6 or on or before D6?"
    # i.e., in row C, columns 1..6 inclusive, or in row D, columns 1..6 inclusive.
    # We only count tiles that are ships in true_board and still hidden in
partial_board.

    def rc_to_index(row_letter: str, col_num: int):
        r = ord(row_letter.upper()) - ord('A')
        c = col_num - 1
        return r, c

    mask = np.zeros_like(true_board, dtype=bool)
    for row_letter, col_num in [('C', 6), ('D', 6)]:
        r, c = rc_to_index(row_letter, col_num)
        # Guard against out-of-bounds in case of different board sizes
        if 0 <= r < true_board.shape[0] and 0 <= c < true_board.shape[1]:
            mask[r, :c+1] = True

    # Unrevealed ship tiles: ship on true_board and not yet revealed on
partial_board
    unrevealed_ship = (true_board > 0) & (partial_board != true_board)
    return bool(np.any(unrevealed_ship & mask))
```

D.2 CaptainQA Evaluations

LLM	Captain Type	F1 Score	Precision	Recall	Moves	Questions	EIG	Redundant Qs
Human	Human	0.615	0.467	0.968	28.325	8.198	0.351	0.028
Baseline	Random	0.317	0.210	0.665	40.000	0.000	-	-
	Greedy	0.614	0.456	0.989	28.778	0.000	-	-
Llama-4-Scout	LM	0.367	0.264	0.630	30.426	14.852	0.242	0.185
	+Bayes-Q	0.393	0.284	0.657	29.352	14.833	0.469	0.002
	+Bayes-M	0.685	0.537	1.000	24.926	14.333	0.266	0.146
	+Bayes-QM	0.741	0.601	1.000	21.667	13.907	0.479	0.001
	+Bayes-QMD	0.764	0.639	1.000	21.000	14.981	0.490	0.000
GPT-4o	LM	0.450	0.308	0.863	36.593	14.056	0.296	0.146
	+Bayes-Q	0.471	0.324	0.901	36.426	14.296	0.476	0.012
	+Bayes-M	0.727	0.583	1.000	22.426	12.037	0.348	0.062
	+Bayes-QM	0.753	0.615	1.000	21.074	11.556	0.501	0.003
	+Bayes-QMD	0.782	0.659	1.000	20.074	14.963	0.513	0.000
GPT-5	LM	0.716	0.568	1.000	22.833	7.981	0.470	0.002
	+Bayes-Q	0.698	0.547	1.000	23.815	7.722	0.499	0.000
	+Bayes-M	0.708	0.561	1.000	23.333	7.963	0.474	0.000
	+Bayes-QM	0.722	0.575	1.000	22.352	7.167	0.499	0.000

Table 4: CaptainQA: Overall results. *F1 Score* (Targeting Score) is the harmonic mean of *Precision* and *Recall* over the board as a binary classification task. *Moves* (out of 40) and *Questions* (out of 15) count the total number of moves and questions asked, respectively. *EIG* ($\varepsilon = 1.0$) measures the average expected information gain of questions asked, with ceiling value ≈ 0.531. *Redundant Qs* is the fraction of questions yield no information gain (i.e., $\text{EIG}_\varepsilon(q) = 0$).

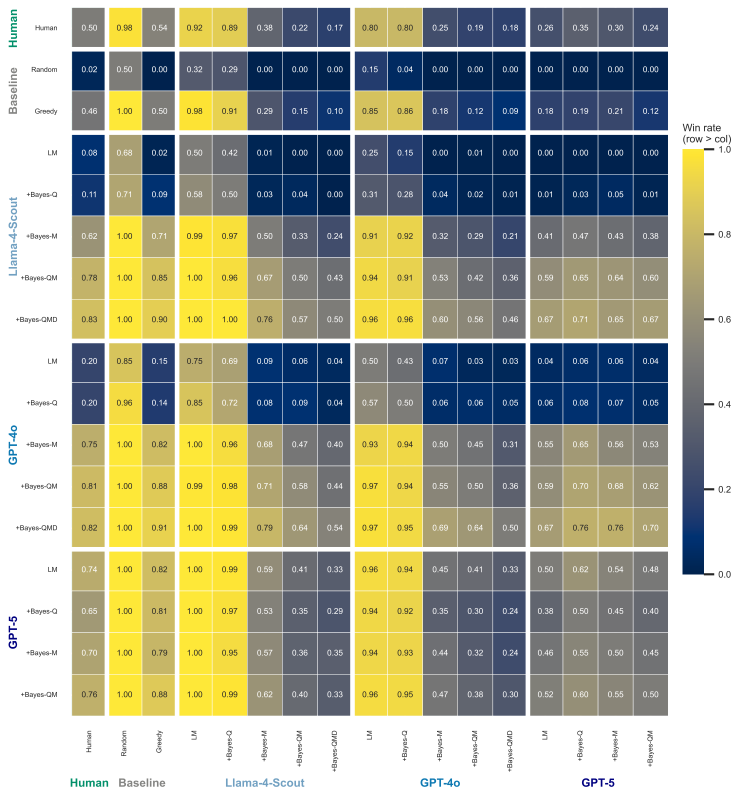


Figure 18: Win rate heatmap between Captain strategies. The "win rate" metric (defined in §4.2) measures board-matched performance in simulated head-to-head play between two Captains. Each cell in the heatmap table shows the win rate of the *row* over the *column*; 0.50 indicates balanced performance, while >0.50 indicates that the row Captain consistently beats the column Captain.

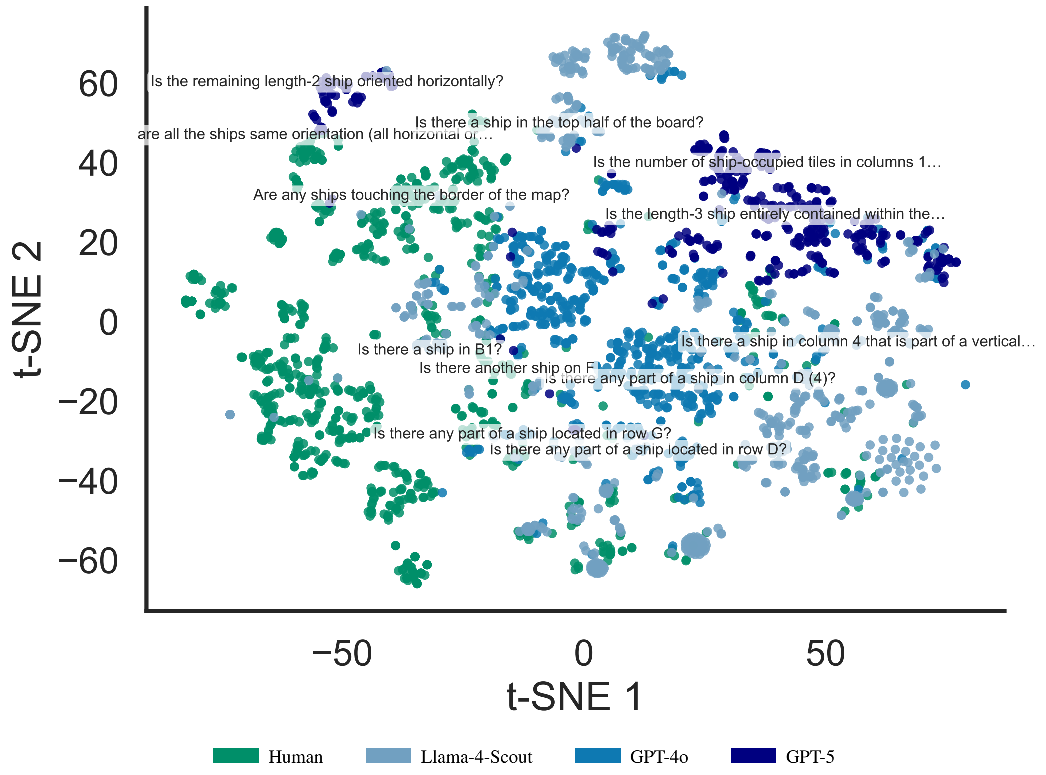


Figure 19: Visualization of the space of human and LM questions. Each point is a question asked during play, with 2D coordinates obtained by applying t-SNE to PCA-reduced question embeddings, obtained from text-embedding-3-small from OpenAI. Colors indicate different LMs. We annotate a small number of representative questions per group, selected via stochastic distance-weighted sampling around per-group PCA centroids. The resulting map highlights that human and model captains occupy overlapping but systematically distinct regions, revealing differences in the kinds of questions they prefer to ask.

LLM	Captain Type	Input Tokens	Output Tokens	Input Cost (USD)	Output Cost (USD)	Total Cost (USD)
Llama-4-Scout						
	LM	6,882,573	2,156,078	0.55	0.65	1.20
	+Bayes-Q	14,211,221	4,533,128	1.14	1.36	2.50
	+Bayes-M	3,751,304	998,736	0.30	0.30	0.60
	+Bayes-QM	10,603,666	3,334,927	0.85	1.00	1.85
	+Bayes-QMD	8,791,089	2,464,043	0.70	0.74	1.44
	Total	44,239,853	13,486,912	3.54	4.05	7.59
GPT-4o						
	LM	7,002,350	1,755,117	17.51	17.55	35.06
	+Bayes-Q	15,620,891	4,244,231	39.05	42.44	81.49
	+Bayes-M	3,520,367	994,233	8.80	9.94	18.74
	+Bayes-QM	9,532,203	2,917,634	23.83	29.18	53.01
	+Bayes-QMD	9,287,099	2,759,823	23.22	27.60	50.82
	Total	44,962,910	12,671,038	112.41	126.71	239.12
GPT-5						
	LM	4,022,670	13,880,517	5.03	138.81	143.83
	+Bayes-Q	8,002,611	35,823,591	10.00	358.24	368.24
	+Bayes-M	2,755,613	7,952,909	3.45	79.53	82.97
	+Bayes-QM	6,074,985	27,092,481	7.59	270.93	278.52
	Total	20,855,879	84,749,498	26.07	847.50	873.57

Table 5: Token usage and dollar cost of agents on the CaptainQA benchmark. We report totals summed across all 54 games for each Captain. Note the difference in pricing across LMs spanning two orders of magnitude: while Llama-4-Scout costs approx. 8 USD total across all experiments, GPT-5 costs approx. 900 USD.

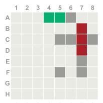
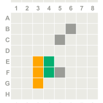
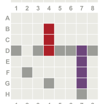
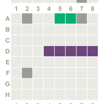
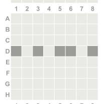
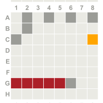
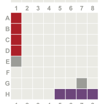
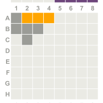
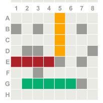
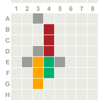
Model	Captain	EIG	Question	Board
Human	–	0.12	one in the bottom left corner?	
	–	0.20	is there a boat in the bottom right corner	
	–	0.42	Is there a horizontal ship in row A?	
	–	0.44	trigger happy lol, is there a ship in F6, F7, G6, or G7?	
	–	0.49	Is the ship on D4?	
	–	0.52	Is orange vert?	
	–	0.26	are there any ships in row 4-5	
	–	0.40	is there any ship in the bottom right	
	–	0.43	is the purple ship surrounded by water	
	–	0.48	upper yes, lower no	

Table 6: Example questions from human Captains. Questions are randomly sampled from the BATTLESHIPQA dataset.

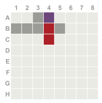
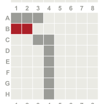
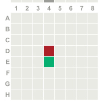
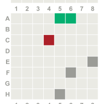
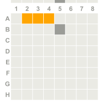
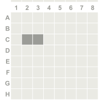
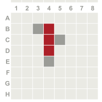
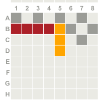
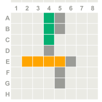
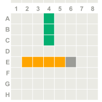
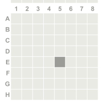
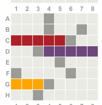
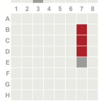
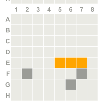
Model	Captain	EIG	Question	Board
Llama-4-Scout	LM	0.00	Is there a ship in row A?	
	+Bayes-Q	0.51	Is there a ship in row C that is part of a vertical ship?	
	+Bayes-M	0.31	Is there a ship in column 5?	
	+Bayes-QM	0.40	Is there a ship in column 4 that extends into row A or B?	
	+Bayes-QMD	0.53	Is there a ship in row B that spans more than two columns and starts in column 2 or further?	
GPT-4o	LM	0.53	Is there at least one ship in the section covering rows B to B and columns 2 to 4?	
	+Bayes-Q	0.52	Is there any part of a ship located in rows F or G and columns 1 to 4?	
	+Bayes-M	0.12	Is there a ship tile in rows E1-H4?	
	+Bayes-QM	0.51	Is any part of a ship present in tiles D1, D2, D3, E7, or H1?	
	+Bayes-QMD	0.53	Does row F contain any part of a ship?	
GPT-5	LM	0.40	Is the length-5 ship on row D?	
	+Bayes-Q	0.37	Does the not-yet-sunk length-2 (green) ship occupy at least one tile in columns 6-8?	
	+Bayes-M	0.18	Is at least one of the remaining length-2 ships entirely contained within columns 1 through 4?	
	+Bayes-QM	0.14	Is the remaining length-3 ship located entirely within the top half (rows A-D)?	

Table 7: Example questions from agents. Questions are randomly sampled from game trajectories from the CaptainQA experiment.

E *Guess Who?* EXPERIMENT

E.1 THE *Guess Who?* GAME

In order to validate our methodological contributions, in this section we discuss an extension of our framework to another setting, a version of the board game *Guess Who?* .

This is typically a two-player game, where both participants are given the same set of 24 human characters, each with a unique combination of given traits (e.g. hair color, eye color, facial hair). Each player selects one character, and players alternate asking each other binary questions (e.g. ‘Is your character wearing a hat?’) to try to determine what character the other player picked. The objective of the game is to be the first player to identify the other player’s character.

In this experiment, we adapt the “Guess-Who-v0-raw” environment from TextArena (Guertler et al., 2025). This is a one-player version of the game, where an LM is tasked with identifying a character picked randomly from a set of 24 in the least amount of moves possible. The LM only gets one guess: if the LM ever guesses the wrong character, the game ends in a loss.

In the default TextArena setting, LMs get a maximum of 40 questions for these 24 characters, meaning trivial strategies such as asking “Is your character X?” for all characters are valid. Thus, to incentivize models to attempt interesting strategies, we restrict the question budget to 8 questions, and increase the total number of characters to 100.

The new characters are sampled by querying GPT-5 with the existing set of characters in JSON format, and asking it to produce 76 more characters like them. The full character list was then validated to ensure no two characters were indistinguishable, and that the game could therefore still be played.

E.2 EXAMPLE CHARACTERS

Two example characters from our set of 100 follow:

```
{
    "name": "Alex",
    "gender": "male",
    "hair_color": "brown",
    "hair_style": "short",
    "eye_color": "brown",
    "accessories": ["glasses"],
    "facial_hair": "mustache",
    "skin_tone": "light",
    "hat_type": "none",
    "hair_texture": "straight",
    "eyewear_style": "round",
    "nose_shape": "pointed",
    "ear_size": "medium",
    "smile_type": "wide",
    "clothing_style": "casual",
    "age_range": "middle-aged",
    "complexion": "fair",
    "cheek_features": "dimples"
},
{
    "name": "Elena",
    "gender": "female",
    "hair_color": "blonde",
    "hair_style": "short",
    "eye_color": "green",
    "accessories": ["scarf"],
    "facial_hair": "none",
    "skin_tone": "fair",
    "hat_type": "none",
    "hair_texture": "straight",
    "eyewear_style": "round",
    "nose_shape": "round",
    "ear_size": "small",
    "smile_type": "wide",
    "clothing_style": "sporty",
    "age_range": "middle-aged",
    "complexion": "olive",
    "cheek_features": "freckles"
  }
```

E.3 Modeling QA Behavior

We then implement the same LM strategies as Section 4.3, which we sum up in Table 8 below.

Note that since the game is only composed of one guess, the 'Decision' function is just a confounding factor: given the information bottleneck, models should just ask every question they can and guess when they run out of questions. This motivates the definition of this setting's decision function $d_{GW}(\mathcal{H}_{1:t})$ below, as well as justifying the removal of the *Bayes-QMD* condition, since there is no rational decision-making ability to model.

This also means the symbolic baselines *Random* and *Greedy* are no longer useful comparisons, as both would immediately make their guess, picking a character with no prior information, and thus obtain a success rate around $\dfrac{1}{100 \text{ characters}} = 1\%$.

Player	Decision	Question	Guess
LM	$d_{\text{GW}}(\mathcal{H}_{1:t})$	$p_{\text{LM}}(\cdot \mid x, \mathcal{H}_{1:t})$	$p_{\text{LM}}(\cdot \mid x, \mathcal{H}_{1:t})$
+ Bayes-Q	$\vdots$	$\arg\max_{q \in \mathcal{Q}} \text{EIG}_{\varepsilon}(q \mid x, \mathcal{H}_{1:t})$	$\vdots$
+ Bayes-M	$\vdots$	$p_{\text{LM}}(\cdot \mid x, \mathcal{H}_{1:t})$	$\arg\max_{i,j} \pi_t(\cdot \mid x, \mathcal{H}_{1:t})$
+ Bayes-QM	$\vdots$	Q_{Bayes}	M_{Bayes}

Table 8: Summary of *Guess Who?* strategies. As in 1, *LM* is a pure language model strategy, which the *Bayes* strategies build upon. Triple-dots indicates inheritance from the row above.

$$d_{\text{GW}}(\mathcal{H}_{1:t}) = \begin{cases} \text{QUESTION}, & \text{If \# Questions Asked} < 8, \\ \text{GUESS}, & \text{Otherwise} \end{cases}$$

Figure 20: The decision function for the *Guess Who?* experiment

E.4 Experimental Details

As with the Battleship CaptainQA experiments (Section 4.3), we test GPT-4o and Llama-4-Scout as our player (equivalent to a "Captain", in our Battleship framework) models, fixing GPT-5 as our gamemaster (a "Spotter") model to answer the player's questions and provide feedback on the final guess. We do not evaluate GPT-5 as a player model due to cost reasons.

We run each of the strategies described in Table 8 for 60 games. We query all models via the OpenRouter API with default parameters.

E.5 Results

A graph showing the results of these experiments is available in the main text as Fig. 5. A more precise breakdown of the results, including the average EIG per condition and the percentage of redundant questions, is available in Table 9 below.

The results in Table 9 show that the Bayesian strategies we propose, originally applied to Battleship, also apply to *Guess Who?* . In both models, success rate massively increases from the base LM condition to *Bayes-QM*: GPT-4o's success rate increases from 0.617 to 0.900, and Llama-4-Scout's

LLM	Captain Type	Success Rate	EIG	Redundant Qs
Llama-4-Scout	LM	0.300	0.413	0.006
	+Bayes-Q	0.450	0.478	0
	+Bayes-M	0.550	0.436	0
	+Bayes-QM	0.724	0.478	0
GPT-4o	LM	0.617	0.454	0
	+Bayes-Q	0.729	0.512	0
	+Bayes-M	0.667	0.455	0
	+Bayes-QM	0.900	0.506	0

Table 9: *Guess Who?* QA: aggregated success rates and breakdown by model and strategy.

success rate increases from 0.300 to 0.724. As with the Battleship experiments, we can therefore see that Bayesian strategies uplift weak LMs past the performance of much stronger pure LM agents.

The middle conditions *Bayes-Q* and *Bayes-M* also separately improve as compared to both base conditions, suggesting that neither base LM optimally integrates information from all previous questions into its guessing strategy (otherwise we'd see no improvement through *Bayes-M*), and that neither model is reliably able to ask the optimal question at any given point during the game.

The *Bayes-Q* and *Bayes-QM* conditions also significantly raise EIG (0.454 $\rightarrow$ 0.506/0.512 for GPT-4o, and 0.413 $\rightarrow$ 0.478/0.478 for Llama-4-Scout). Redundant (EIG = 0) questions are already rare – GPT-4o never asks any such question in any condition, Llama-4-Scout only asks then 0.6% of the time in the base LM condition – but no such questions are asked in any condition that optimizes for EIG. This is encouraging, since it means our EIG-based reranking approach is finding better questions than the first question the LM considers.

In general, these results point to the generality of this framework, having brought substantial improvements to the performance of LM agents in two very different information-seeking domains.

E.6 *Guess Who?* PROMPTS

E.6.1 SYSTEM PROMPT

System

```
You are playing Guess Who. The other player has selected a character from the list
below, and your goal is to guess which character they have selected by asking
yes-or-no questions about the character's traits.

You can ask 8 questions in total, and you have to make your guess after asking all
of them. If you guess correctly, you win the game. If you guess incorrectly, you
lose the game.

Given this game history:

<history>
{history}
</history>

This is the list of characters you can ask questions about:

<characters>
{characters}
</characters>
```

This prompt is referred to as "{context}" when mentioned in other prompts.

E.6.2 Player Prompts

Guess

```
{{context}}

Your task is to make your one and only guess about the character. Make sure you
consider the context of the theme and all previous questions and answers.

Guess from the list above.

Please think about your answer step by step. When you have come up with a final
answer, respond with your guess wrapped in <answer></answer> tags, and optionally
square brackets, e.g. <answer>Mike</answer> or <answer>[Mike]</answer>
```

Non-EIG Question

```
{{context}}

Your task is to ask a single question that will help you gain the most information
possible about the secret word. You can ask any question, but is must be answerable
with a Boolean answer (yes/no). Make sure your questions are clear, specific and
different from ones you have asked previously, to avoid pigeonholing a potentially
wrong answer too quickly.

You have {{remaining_questions}} questions left, including this one.

Please think about your answer step by step. When you have come up with your
question, please wrap it in <answer></answer> tags: e.g. <answer>Is the character
male?</answer>
```

EIG Question

```
{{context}}

Your task is to generate {{k}} question(s) that will help you gain the most
information possible about the secret word. Each question must be answerable with a
Boolean answer (yes/no).

You have {{remaining_questions}} batches of {{k}} question(s) left, including this
one.

Please think about your answer step by step. When you have come up with your
question, please return your question(s) as a JSON dictionary with numbered keys,
wrapped in <answer></answer> tags like this: <answer>{{"1": "Is the character
male?", "2": "Do they have a mustache?", "3": "Do they have a hat?"}}</answer>

IMPORTANT: Use proper JSON format with double quotes around both keys and values.
```

E.6.3 Game/Sampler Prompts

Consistency

```
You are playing Guess Who. Your task is to determine which characters from the list
below are consistent with the constraint implied by the question.
```

```
The most recent question is "{{question}}".

Given the following characters:

<characters>
{{characters}}
</characters>

Respond with a JSON dictionary wrapped in <answer></answer> tags where the keys are
the characters and the values are either "yes", if the character satisfies the
constraint given in the question, or "no" if they do not.

<answer>{{"character1": "yes", "character2": "no", "character3": "yes"}}</answer>

IMPORTANT: Use proper JSON format with double quotes around both keys and values.
```