

多回合谜题：评估大型语言模型中的交互推理和战略性对话

Kartikeya Badola¹, Jonathan Simon¹, Arian Hosseini¹,

Sara Marie Mc Carthy¹, Tsendsuren Munkhdalai¹, Abhimanyu Goyal¹, Tomáš Kočiský¹, Shyam Upadhyay¹,

Bahare Fatemi² and Mehran Kazemi¹

¹Google DeepMind, ²Google Research

大语言模型 (LLMs) 擅长解决陈述明确且完整的问题，但常常在微妙的环境或互动任务中遇到困难，而这些在大多数现实世界情境中很常见。这凸显了开发能够有效参与逻辑一致的多轮对话、寻求信息和在不完备数据中进行推理的 LLMs 的关键需求。为此，我们引入了一个新颖的基准，其包含一组多轮任务，每项任务都专为测试特定的推理、互动对话和信息寻求能力而设计。这些任务有确定性的评分机制，从而消除了对人工干预的需求。在我们的基准上评估前沿模型揭示了显著的提升空间。我们的分析显示，大多数错误源于指令遵循不力、推理失败和计划不足。该基准提供了关于当前 LLMs 在处理复杂、互动场景时的优缺点的宝贵见解，并提供了一个坚实的平台，供未来研究致力于改善这些关键能力。

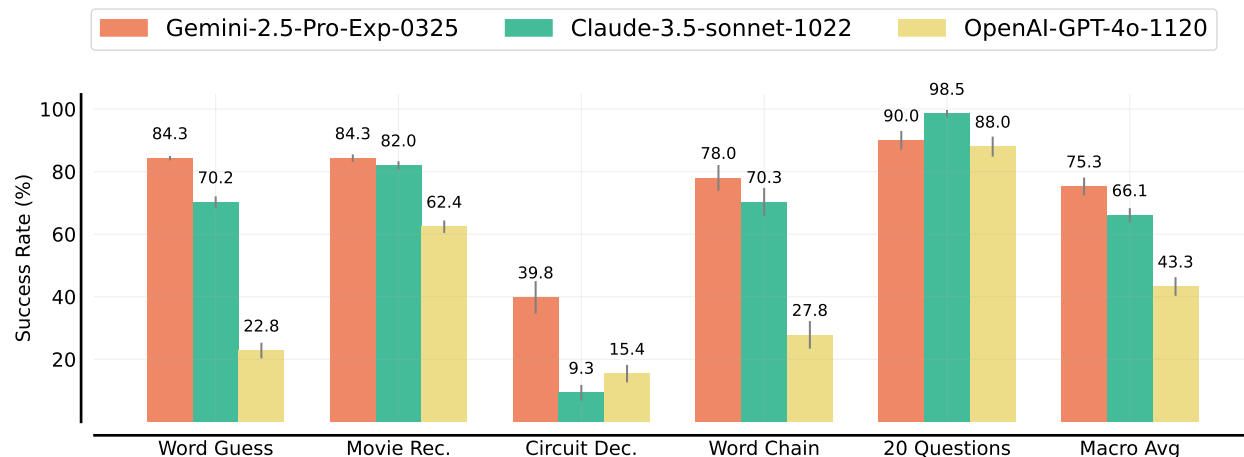


Figure 1 | 多轮谜题 (MTP) 基准：Gemini 2.5-Pro-Exp、Claude-3.5-sonnet 和 OpenAI-GPT-4o 在五个任务中的表现。虽然模型在二十问题上表现良好，其他任务仍有显著的提升空间。

1. 介绍

在信息不完整或难以提取的情况下，大型语言模型 (LLM) 代理经常需要在多轮互动中主动提问以获取更多信息，减少不确定性，并有效解决问题。这需要精心策划、交互的战略计划和推理，以决定在多轮对话中的下一步行动，以及对之前互动保持强大的记忆。此能力对于 LLM 在复杂的真实环境中有效运作是至关重要的，因为在这些环境中信息并不总是通过单一查询清晰地提供，例如虚拟助手和编码助手 (?)。虽然 LLM 代理在单轮问题（例如涉及单次提示的数学和编码问题 (????)）方面表现出色，但当用户的指示不清晰或不充分时，它们通常表现不佳，并且在提供有用的响应之前必须从用户那里获得更多信息。这是一个根本性的问题，需要对 LLM 如何在不完整的信息中进行推理、跨轮次保持一致且合逻辑、并展示信息搜索行为以有限互动进行有效计划和推理的研究和发展。为了评估 LLM 的多轮推理和指令遵循能力，我们提出多轮谜题，这是一个包含五个不同任务的新基准，每个任务都有独特的挑战和可验证的指标 (Table 1)：

- 单词猜测测试模型演绎推理的效率以及参与同质轮流行为的能力。

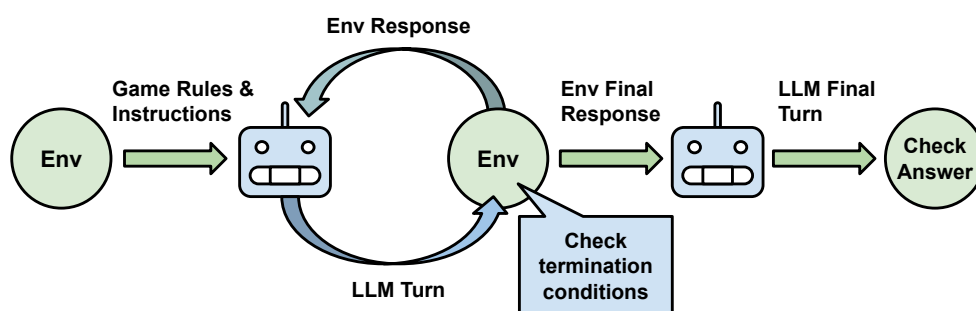


Figure 2 | 多轮交互：LLM 在多轮谜题基准中与环境进行交互，并使用基于规则的方法检查最终输出。

- 电影推荐评估模型在信息不完美场景中寻求信息的能力，并通过互动提问学习用户偏好。
- 电路解码测试逻辑推理以及模型通过多次输入和观察输出来学习新工具内部工作原理的能力。
- 词链探测模型的能力，即在多轮对称交互中保留和确定性地依赖所获取的信息的能力。
- 二十问注重回答查询时在对话中的逻辑一致性和连贯性。

这些任务共同提供了对大型语言模型多轮推理和交互式问题解决能力的全面评估，提供了关于如何训练自主代理在模糊情况下表现得更智能的洞察。我们对我们的多轮任务进行了前沿模型的基准测试，包括 OpenAI GPT-4o、Anthropic Claude 3.5 Sonnet 和 Gemini 2.5 Pro Exp。Figure 1 表明，尽管已经取得了显著进展，但对多轮推理和信息搜索行为的复杂性仍需进一步探索。我们相信这为社区协同推进该领域的最新技术提供了宝贵的机会。

Name	Short Description	Metric
Word Guess	Guess the secret word in min attempts. Environment gives feedback on how close the guess is at each turn.	Normalized number of attempts to solve the task.
Movie Rec.	Probe the user to decode the user preference function for N turns. Pick a movie for the same user at N+1 turn.	Normalized rank of the final movie.
Circuit Dec.	Probe the C different boolean circuits for N turns. Predict the joint truth table of all the circuits at N+1 turn.	Normalized circuit-wise accuracy.
Word Chaining	Model and user take turns choosing allowed words that start with the last letter of the previously chosen word. The episode ends successfully if no words with the correct starting letter remain, or max_turns is hit.	Normalized % of trajectories with successful end state.
Twenty Questions	Model chooses a secret word. The user asks questions to determine what the word is. The model must answer these questions in a way that doesn't contradict previously-provided answers.	Normalized % of logically consistent trajectories.

Table 1 | 多轮谜题任务描述和指标

2. 多回合谜题 (MTP) 基准

在信息不完整的领域通过多轮对话进行推理是人类智能的核心能力，但现有的基准测试未能充分评估这一能力。大多数流行的单轮评估往往未能捕捉真实世界互动的复杂性，其中问题解决常常涉及迭代提问、假设改进和对不断变化的信息做出动态调整。我们的目标是创建一个专门针对这一空白的基准测试。MTP 包含五个不同的任务，包括猜词、电影推荐、电路解码、连词和二十个问题。这些任务旨在评估模型的以下能力：(1) 用部分信息进行推理，(2) 在不同回合中保持逻辑一致性，(3) 展现信息搜寻行为。

2.1. 数据创建与度量

我们的所有任务都是合成创建的（参见 Figure 4 中的示例）。它们利用基于规则的环境和确定性的评分者，消除了对人工参与或自评器的需求，并确保了紧密的置信区间。Figure 2 展示了我们基准测试中多轮互动的概述。本节简要介绍了我们的任务构建。

猜词 目标是在尽可能少的尝试中猜出秘密单词。我们从 10 个包含 40 个单词的独特词汇开始。每个单词有 5 个字母。由于每个单词可以作为一个数据点中的秘密单词，因此总共有 400 种配置。A.3 节概述了该任务的完整说明。

电影推荐 目标是在多轮对话中发现用户的偏好函数，并最终为该用户选择一部电影。每个数据点都有一个模拟用户，具有唯一的偏好函数、电影的已观看集合（用于提问阶段）和未观看集合（用于推荐阶段）。共有 20 个独特用户和 50 种电影集合变化，总计产生 1000 种任务配置。A.1 节描述了用户和电影集合的初始化方式。A.3 节概述了此任务的说明细节。

电路解码 在这个任务中，目标是确定一组 C 个未知布尔电路的结构。每个电路接收 k 个布尔输入并产生一个单比特输出。这些 C 个电路总共由固定数量的与门 (AND)、或门 (OR) 和非门 (NOT) 组成。模型必须推断所有 C 个电路的功能，最终生成它们组合的联合真值表。对于评估集，我们保持 $C \times 2^k = 24$ ，并构建了 300 种独特的电路配置。更多细节在第 ?? 节讨论。

单词接龙 在每个游戏中，通过从 SCOWL 单词列表中随机抽取 500 个单词来构建词汇表 (?)。起始玩家是随机选定的，玩家轮流从词汇表中选择单词，遵循以下规则：(i) 单词必须以上一个单词的最后一个字母开始，并且 (ii) 单词不能是之前使用过的。违反规则会导致立即失败。如果没有有效的动作可选或达到轮次限制，游戏在不失败的情况下结束。LLM 的表现通过其成功率（无失败完成的游戏比例）来衡量。

二十个问题 评估 LLM 在对话过程中保持逻辑一致性的能力。LLM 从一个预定义的 WordNet 词库 (?) (80-100 个单词，提供上位词) 中选择一个秘密名词。然后，它必须回答用户就秘密词或其上位词（包括传递上位词）提出的多达 20 个二元（是/否）问题。只有当且仅当 LLM 提供的答案与其在同一个游戏实例中的先前回答逻辑上不一致时，才会发生失误。

2.2. 回合数和剧集数

Table 2 展示了每个任务每轮允许的最大回合数，以及在评估期间 Gemini 2.5 Pro Exp、Claude 3.5 Sonnet 和 GPT-4o 执行的平均回合数。谜题的最大回合数从电影推荐的 11（10 个交互回合加 1 个最终预测回合）到猜词游戏的 40。谜题具有提前停止条件，这使得游戏可能在达到最大回合数之前终止。这可能发生在诸如模型犯了不可恢复的错误（例如，二十个问题中的逻辑矛盾）或没有可用的有效动作（例如，在文字串接中用尽可玩的词）等情况下。

Task	Num Problems	Max Num Turns	Gemini 2.5	Claude 3.5	GPT-4o
Word Guess	400	40	2.4	3.8	12.9
Movie Rec.	1000	11 (10 + 1)	11	11	10.9
Circuit Dec.	300	19 (18 + 1)	10.5	19	18.4
Word Chaining	400	20	9	9.3	5.1
Twenty Questions	400	21 (20 + 1)	6.3	6.2	15.1

Table 2 | 多轮谜题的统计及前沿模型的平均游戏回合数。游戏包含提前终止条件，使得在回合尚未达到最大值时就有可能结束。这种情况可能发生在模型在二十个问题中犯了逻辑矛盾，或在单词接龙中没有可用单词的情况下。

3. 实验

我们报告了 Gemini-2.5-Pro-Exp-0325、Claude-3.5-sonnet-1022 和 OpenAI-GPT-4o-1120 在我们的基准测试中的表现。对于每个实验，我们使用每个 API 提供的默认采样参数。

Figure 1 展示了这些模型在五个不同任务中的表现。结果表明，虽然模型在二十个问题任务中表现良好，但在其他任务中仍有显著的改进空间。为此，我们通过游戏持续时间、成本相关的表现、模型间轮流以及信息寻求策略来探查模型的行为。

3.1. 模型需要多少次迭代？

像电影推荐、电路解码和二十个问题等任务有定义的交互限制，随后有一个最终的动作，这反映在它们的最大回合数（例如，分别为 10+1, 18+1, 20+1）。值得注意的是，Gemini 2.5 使用的平均回合数表明，一些任务（Table 2），如电路解码（平均 18.85 回合），通常需要接近全部的交互预算，而其他任务如猜词（平均 5.37 回合）通常较快完成。

Figure 5 显示了各模型在我们的任务中进行的回合数量的分布。在电影推荐任务中，所有模型的分布都紧密集中在允许的最高 11 回合（10 次互动 + 1 次最终推荐）附近，反映了任务设计的特点，即需要利用全部互动预算来收集用户偏好，然后进行最终推荐。相反，在猜单词任务中，各模型的中位数回合明显较少，尤其是 Gemini 2.5 和 Claude 3.5，表明模型通常能够在宽松的 40 回合限制之前高效地发现秘密单词。电路解码、词链和二十问等任务表现出更大的变异性。电路解码中，Claude 3.5 和 GPT-4o 一致使用最多的 19 回合，而 Gemini 2.5 则显示了更广泛的分布，这反过来在他们该任务的成功率中显得明显。词链显示了显著的扩展，反映出策略深度的差异。二十问不同的分布显示了保持逻辑一致性的时长。总体而言，回合分布有效地展示了任务约束（如固定互动长度）和模型特定策略如何互动以决定对话长度。

3.2. 成本-性能权衡

最近的研究表明，与较大的模型相比，成本效益模型的推理方式存在系统性差异 (?)。基于此，我们进一步扩展分析，评估这些更实惠的模型在我们的一组多轮互动任务中表现如何。我们评估了 Gemini 2.0 系列中的不同变体——Pro-Exp、Flash 和 Flash-lite——这些变体在计算成本方面代表了不同的层次，其中 Pro 是最昂贵的，而 Flash-lite 是最便宜的。

在 Figure 6 中的结果揭示了成本与性能之间的明显权衡。Gemini 2.0 Pro 在所有任务中始终达到最高的成功率，其次是 Gemini Flash，然后是 Gemini Flash-lite。这一趋势在宏观平均分中表现得尤为明显。性能差距根据任务的不同而变化；例如，在电路解码任务中差距尤为明显（Pro: 32.2%、Flash: 7.2%、Lite: 4.1%），而在诸如“二十个问题”这样的任务中差距相对较小，但仍然存在（Pro: 95.5%、Flash: 91.2%、Lite: 82.5%）。

3.3. 较弱模型和较强模型能否完成彼此的思路？

模型经常可能会被提示提供不准确或不完整的对话或推理背景。对于模型来说，能够检测提供的历史信息中的缺陷、提取有用信息以及掌握所用策略是很重要的。在这方面，我们研究了一个较强的模型（Gemini 2.5 Pro）是否可以有效地结束由一个较弱的模型（Gemini 2.0 Flash）发起的对话，反之亦然。该实验涉及一个模型处理初始交互（ $n-1$ ），然后由另一个模型来进行最后一轮交互。我们在电影推荐任务上进行此实验，该任务在中间轮次（偏好探查）和最后一轮（推荐）之间表现出强烈的非对称性。

当较弱的 2.0 Flash 模型完成 2.5 Pro 模型的互动时，得分达到了 75.2 %。令人惊讶的是，当较强的 2.5 Pro 模型完成由 2.0 Flash 模型开始的对话时，成功率相似，为 75.7 %。值得注意的是，用较弱的模型进行最终回合导致性能下降了 11 %（相对于较强模型的基线），而允许较强的模型完成则相比于较弱模型的基线取得了 13 % 的提升。这突显了在不对称任务中最后推理步骤的影响，表明一个能力强的模型在执行最后一回合时，可以显著减轻较早之前不太有效的互动回合所带来的缺陷。

我们进行了定性分析，比较在电影推荐任务中的一个更强大的模型（Gemini 2.5 Pro）和一个能力较弱的模型（Gemini 2.0 Flash-Lite）。我们的目标是了解它们在推理和规划方法上的差异，并找出较弱和较强模型之间性能差距的主要原因。我们的分析揭示了以下关键区别，图 7 中展示了来自两个模型的对话示例。

推理策略：能力较低的模型采用了较简单的推理策略，提问时一次仅关注一个电影属性，并根据这些单个回答对那些属性做出明确的结论。这些问题常常显得无关，并未形成一个一致的计划，后续问题并未基于从以前问题中获得的信息进行构建。相比之下，能力更强的模型提出的问题同时考虑多个特征，使用户的回答能提供更丰富的见解。这些问题是相互关联的，并形成了一个更连贯的计划，模型根据早期互动中的观察设计其下一个问题。这包括当新信息与之前的结论相矛盾时重新评估之前的结论，并进一步调查。图 7 中提供的示例说明了这些策略的不同之处。

提出有效的问题：当模型决定寻求特定信息时，我们观察到能力较弱的模型有时会提出无效的问题以获取该信息。例如，如图 7 所示，该模型可能希望了解用户对“原声带存在”的偏好，但接着提出一个关于两部都在此特征上得分很高的电影的比较问题。相反，能力较强的模型通常会比较在其他大多数方面相似而仅在其想要探讨的特定特征上有所不同的电影。

得出准确的结论：我们发现，一些由能力较弱的模型得出的结论是不准确的或缺乏深度。例如，在图 7 中，我们看到该模型在最初的问题之后，草率地对用户的喜好和厌恶做出了许多肤浅的假设。同样地，在比较在多个特征上存在差异的两部电影时，该模型可能仅根据单一特征得出结论，而不考虑其他不同特征的影响。然而，能力较强的模型往往能从收到的答案中得出更有意义的结论，并考虑各种因素的相互作用，这些因素可能解释用户的反馈。

策略性提问：理想情况下，模型提出的每个问题都应有助于收集相关信息，从而帮助做出最终推荐。然而，对于能力较弱的模型，我们观察到某些情况下问题并没有特别有用。例如，图 7 中的第一个问题仅仅比较了提供表格中的前两部电影，其结果信息并没有显著促进后续的决策过程。此外，最后的问题似乎主要是为了满足任务要求，因为模型在第九个问题后已经做出了决定。虽然对更强模型所提问题的最优性进行明确评估是具有挑战性的，但我们观察到，它一直在探寻更多信息，以帮助推进任务。

4. 相关工作

多轮交互中的挑战。 大型语言模型的能力引发了显著的研究兴趣，最初主要集中在单轮任务上。近期研究强调了大型语言模型在从单轮成功过渡到有效多轮互动时面临的挑战。[?] 将这种对话任务（如回忆和扩展）能力下降归因于相关上下文的距离和错误传播等因素。[?] 特别在数学问题解决领域中回应了这些发现，指出在需要持续推理的互动设置中，相较于单轮数学问题，性能出现下降。[?]

引入了 Multi-IF 基准来评估多轮和多语言指令遵循情况，发现每轮都存在性能下降，尤其是在非拉丁文脚本方面存在特殊困难。这些研究强调了更好评估方法的必要性，因此我们设计了我们自己的任务，使其只能通过逻辑和互动对话来解决。

评估多轮对话和推理。 不同的基准测试旨在评估多轮对话的能力。？评估大型语言模型 (LLMs) 生成人类风格多轮聊天对话的能力，并使用其他强大的大型语言模型如 GPT-4 (?) 作为评委。类似地，MT-Bench 和 Chatbot Arena 都评估大型语言模型的多轮交互质量 (?)。MT-Bench 使用一个多轮提问集和基于大型语言模型的评分，而 Chatbot Arena 则依赖于来自真实用户交互的成对比较。？使用一个实体推理游戏，其中模型通过提问来探查对话推理和计划。我们的工作通过像“二十个问题”这样的任务进行补充，在这些任务中，模型回答问题，以测试其逻辑一致性。与这些关注总体质量或指令遵循的方法相比，我们专门评估大型语言模型在谜题场景中多轮推理和信息寻找能力，使用基于规则的可验证指标。

自主行为与交互环境 诸如 LMRL Gym (?) 和 AgentBench (?) 等平台在互动环境中对 LLM 作为代理进行评估。LMRL-Gym 聚焦于增强学习 (RL) 能力 (例如规划、信用分配)，在像“猜我的城市”这样的游戏中，而 AgentBench 在模拟真实世界环境中 (如数据库和网络浏览 (??)) 评估指令执行、推理和知识获取。相较之下，我们的基准提供了更简单的可控游戏环境，允许对多轮能力进行更有针对性的评估。此外，许多上述游戏需要 LLM 模拟游戏环境和用户互动，而我们提供基于规则的环境。？探索针对代码生成的多轮提示和微调，分享我们对多轮交互的关注。然而，我们的谜题环境独特地需要逻辑推理、战略规划和有效沟通。其他相关工作包括基于文本的游戏引擎，例如 TextWorld (?) 和基于这些引擎开发的游戏，如 Coin Collector (?) 和 Jericho-QA (?)，主要用于评估 RL 算法和代理行为。?? 探索上下文中的 RL，重点关注在部分可观察环境中的探索。？评估其他 LLM 能力，例如科学推理和道德，特别是在多轮环境中。有关 LLM 的多轮交互能力的全面调查和详细信息，请参阅？。MTP 不同于依赖 LLM 评审或广泛关注复杂模拟中的代理能力的基准测试，它使用更简单、可控的游戏环境，具有确定性、基于规则的评分。这允许对核心互动推理技能进行有针对性的评估，与单纯的会话质量或指令执行不同。尽管 LLMs 在单回合任务上表现出色，但现实世界的应用往往需要持续的互动、多回合推理和对话能力。我们的工作引入了多回合谜题 (MTP) 基准，通过将互动式解谜作为这些复杂推理和对话规划技能的特定测试平台来贡献这一领域。在 MTP 上评估当前前沿模型显示出各任务之间显著的性能差异，突显了巨大的改进空间，特别是在高效的信息寻求 (电影推荐、猜词)、复杂的推理 (电路解码) 和策略互动 (单词连锁) 方面，尽管在逻辑一致性 (二十个问题) 方面表现较强。定性分析进一步指出了与推理策略、问题表述、结论准确性和规划相关的关键失误模式。MTP 基准，凭借其基于规则的环境和确定性指标，提供了一个强大的平台来诊断这些弱点。未来的工作应利用这些洞见，进一步研究如何微调模型以实现更有效的多回合推理和连贯、战略性的对话，从而最终提高其在现实世界交互中驾驭复杂性的能力。

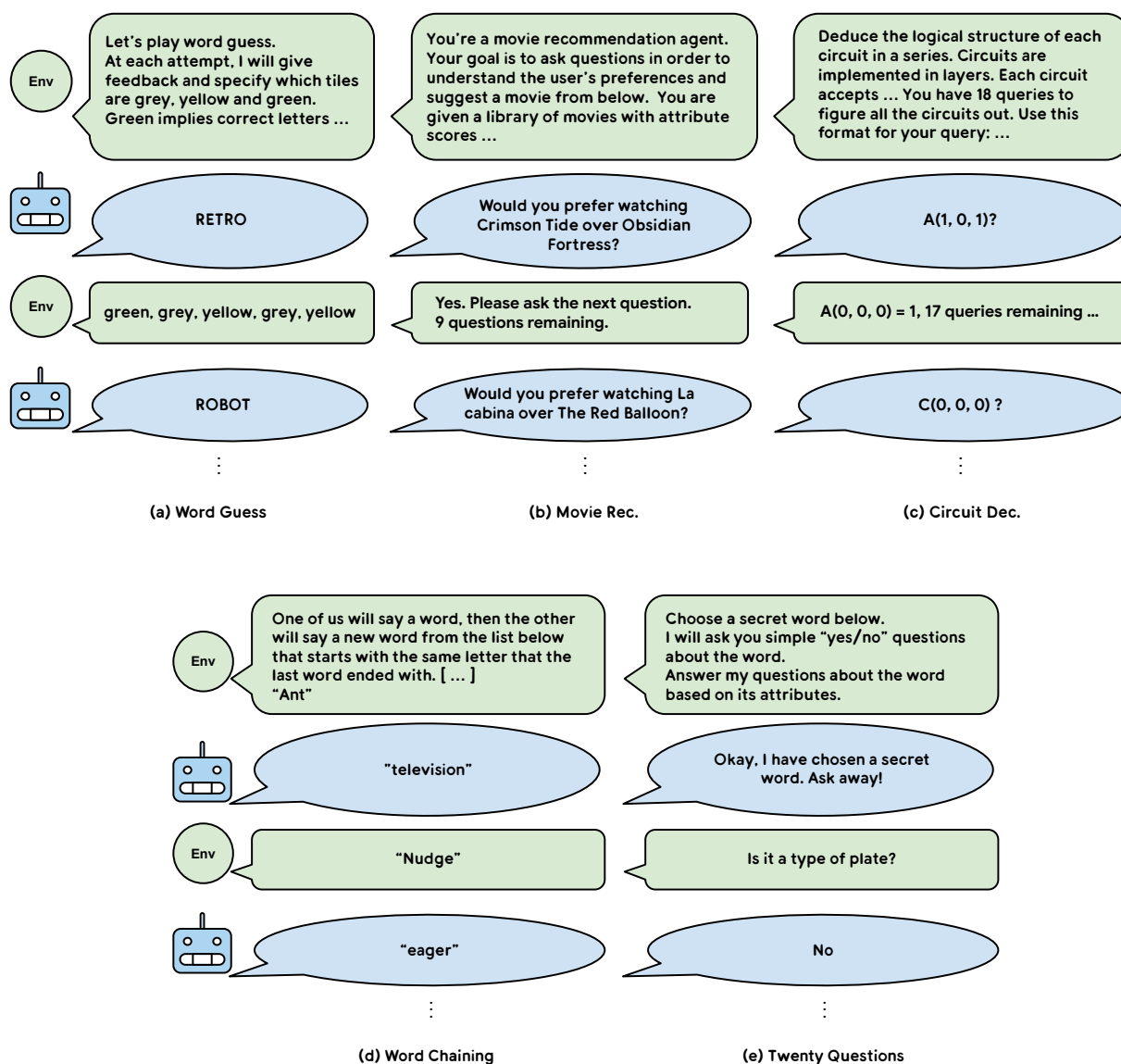


Figure 3 | 多轮谜题任务示例。(a) 猜词游戏：模型尝试猜测一个秘密词语，在每次猜测后获得关于字母正确性和位置的反馈。(b) 电影推荐：模型通过交互式提问来了解用户偏好，然后推荐电影。(c) 电路解码：模型通过对布尔电路进行不同输入的查询来推断其逻辑结构。(d) 单词接龙：模型和环境轮流说出某个单词列表中的单词，每个新单词必须以前一个单词的最后一个字母开头。(e) 二十个问题：模型选择一个秘密词语，必须以逻辑一致的方式回答用户的是/否问题。

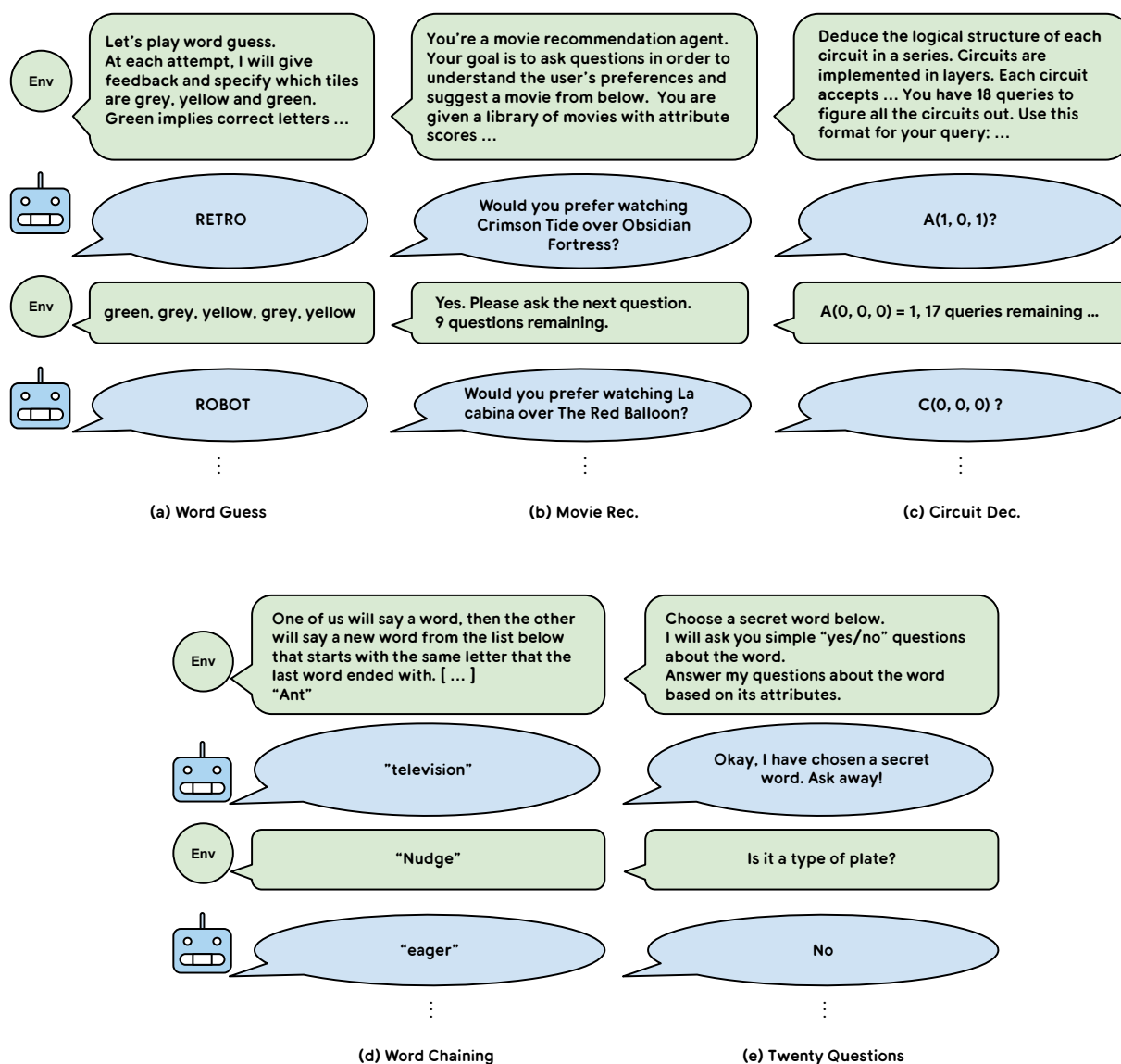


Figure 4 | 多轮益智任务的例子。(a) 单词猜测：模型尝试猜测一个秘密单词，在每次猜测后获取关于字母正确性和位置的反馈。(b) 电影推荐：模型通过交互式提问以了解用户偏好，然后推荐电影。(c) 电路解码：模型通过使用不同输入来查询布尔电路，以推断其逻辑结构。(d) 单词接龙：模型和环境轮流从一个列表中说单词，每个新单词必须以前一个单词的最后一个字母开头。(e) 二十个问题：模型选择一个秘密单词，必须以逻辑一致的方式回答用户的“是/否”问题。

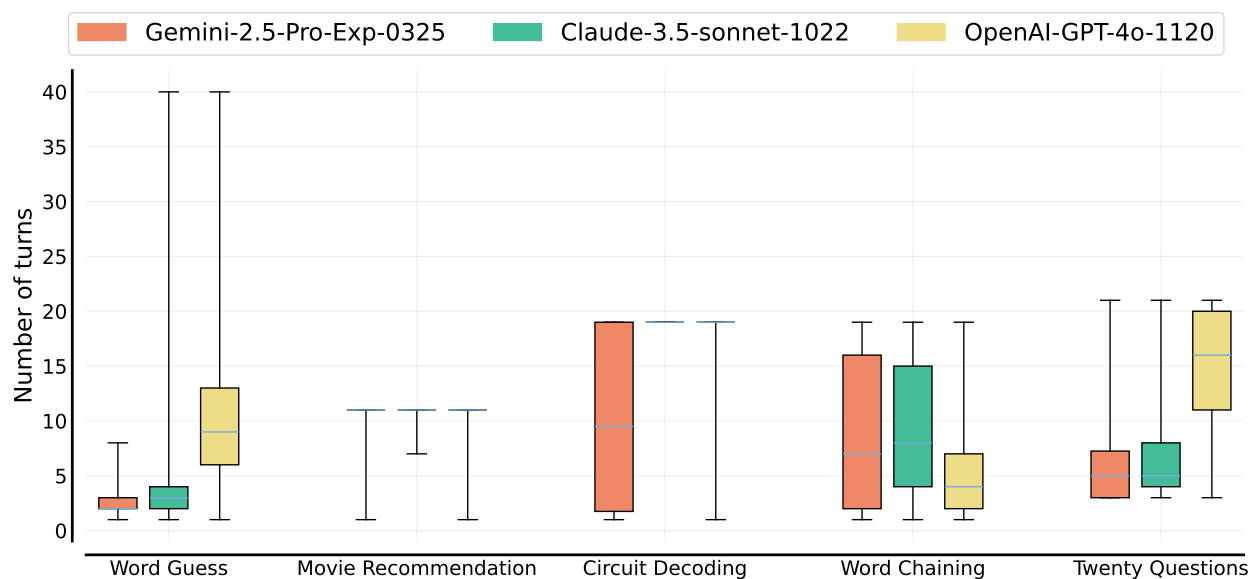


Figure 5 | 模型所采取的轮次分布。同一种游戏类型可能由于达到早期终止条件以决定胜负而有不同的轮次数。任务约束（如固定的交互长度）和特定于模型的策略共同影响对话轮次的数量。

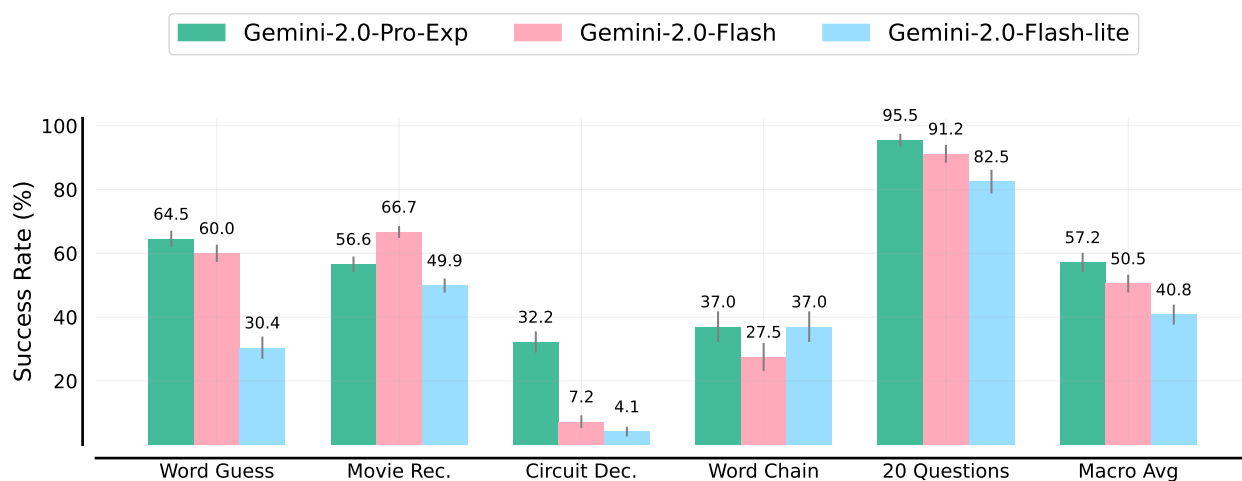


Figure 6 | 结果表明在成本和性能之间存在明显的权衡，较大的 Pro 型号在所有任务和宏平均上都比更具成本效益的 Flash 和 Flash-lite 变体取得了更高的成功率。

Gemini 2.0 Flash-Lite	Gemini 2.5 Pro
[Q1: Simply compares the first two movies]	[Q1: Compares two movies that are mostly similar, but one has high "Intellectual Depth" and one has high "Pace".]
Okay, based on the response, the user seems to appreciate movies that have more "Intellectual Depth", "Visual Details", "Realism Level" and "Emotional Intensity" and less "Pace", "Dialogue Focus", "Soundtrack Presence", "Character Complexity". [Many superficial conclusions] I will confirm my understanding with another question. [Q2: Compares two movies one high in realism and one high in dialogue focus]	This gives me the first piece of information about your preferences. [Q2: Compares two movies that are mostly similar, but one has high "Intellectual Depth" and one has high "Pace" and high "Emotional Intensity".]
Okay, the user clearly doesn't like "Pace" and "Dialogue Focus". Let's try to understand if the user cares about "Emotional Intensity" [wrong conclusion: the winner had a much higher dialogue focus.] [Q3: Compare two movies one high and one low in emotional intensity]	[...] Your preference might lean towards Intellectual Depth over the other two [Pace and Emotional Intensity]. Let's test Emotional Intensity more directly. [Q3: Compares two movies, one has much higher "Emotional Intensity", the other has slightly higher values for several attributes.]
The user seems to like "Emotional Intensity". Now let's understand the importance of "Visual Details". [Q4: Compare two movies one high and one low in visual details]	[...] Both movies have the same Intellectual Depth and Pace. X has much higher Emotional Intensity, while Y has higher Dialogue Focus, Visual Details, and Realism Level. Your preference suggests that Emotional Intensity might be a significant positive factor for you, potentially outweighing Dialogue Focus, Visual Details and Realism. Let's test this further by comparing X against a movie with even higher Emotional Intensity but lower Pace.
[...] Now, I will try to evaluate how much the user cares about "Character Complexity".	[...]
[...] I am going to check how the user values "Soundtrack Presence". [Wrong probe: Compares two movies both high in soundtrack presence]	[...] This might suggest that Intellectual Depth or Soundtrack Presence are quite important, or perhaps your preference regarding Pace or Emotional Intensity isn't as straightforward as previously thought. [Re-evaluating previous conclusions in face of new observations.] Let's try to isolate the effect of Soundtrack Presence versus Character Complexity and Realism Level.
The user doesn't like "Soundtrack Presence" much [wrong conclusion]. Let's explore the importance of "Realism Level".	[...]
[...]	This is very informative. It strongly suggests that you highly value Character Complexity and Realism Level [...]. It also indicates that you likely dislike high Dialogue Focus, as "The Clockwork Mind" has a very high score in that attribute. This aligns well with previous observations about Character Complexity and Realism Level being important positive factors. For my final question, let's try to clarify the relative importance of Emotional Intensity versus Pace and Intellectual Depth.

Figure 7 | 样本（大幅简化）模型对 Gemini 2.0 Flash-Lite（左）和 Gemini 2.5 Pro（右）的响应，展示了它们在电影推荐中的推理和计划过程（为简洁起见，仅显示模型响应）。提供在 [] 中的文本是作者提供的额外信息和背景。

A. 任务创建详情

A.1. 电影推荐

1. 用户：

- 用户偏好函数：我们创建一个用户群体，每个用户都有自己独特的偏好函数，其模型为电影属性的线性函数：

$$\text{score} = w_1 \cdot \text{attribute}_1 + w_2 \cdot \text{attribute}_2 + \dots + w_k \cdot \text{attribute}_k$$

其中 w_i 是权重， attribute_i 是电影属性分数。权重 (w_i) 为每个用户随机生成，其中 $w_i \in [0, 1]$ ，并四舍五入到小数点后一位。

2. 电影：

- 预算约束：为了避免所有属性得分都很高的异常电影，我们给每部电影分配一个预算，限制总属性评分：

$$\text{movie_budget} = \sum_{i=1}^k \text{attribute_score}_i$$

，其中 k 是属性的数量。 movie_budget 在 $(\text{base_budget}, \text{base_budget} + \text{budget_delta})$ 之间随机抽样。

- 最低学分：每个属性都被分配了一个最低分数 ($\text{attribute_min_credit}$)。
- 随机分配：剩余的预算（在分配最低学分后）在属性之间随机分配，以确保电影资料的多样性。
- 已见集合与未见集合：每一集都有两组电影：
 - 可见集：用于代理的质询阶段， $n=20$
 - * 属性分数的精度较低（小数位数较少）。
 - * 预先决定的一部分属性可能会将其得分设置为 min_value 。这使得代理人更容易比较仅在少数属性上有所不同的电影。
 - 未观察集：用于最终推荐， $n=40$
 - * 属性得分的更高精度。
 - * 不像已见集那样有任何稀疏性

该过程持续进行，直到该电路的所有分配门都被使用完毕，从而产生单一输出信号。

A.2. 二十个问题

- 游戏进程：模型被要求从一个固定的词集及其对应的 Wordnet 上位词（即属性）中选择一个秘密词。用户接着询问模型关于所选秘密词属性的一系列是/否问题（例如，“它是一种哺乳动物吗？”），模型需要对此作出回应。如果模型在任何时候作出的回应与之前给出的回答在逻辑上矛盾，游戏立即结束，模型失败。否则，游戏继续进行，直到用户最后猜测秘密词的身份（例如，“你的词是吉娃娃吗？”），模型对此进行回应。模型对于此最终问题的回应再次接受逻辑一致性的评估，如果不一致，游戏以失败告终，否则游戏以成功结束。用户在以下任一情况下尝试最后猜测：(1) 仅剩一个合理一致的词，(2) 达到最大回合数，或 (3) 发生随机提前猜测的情况。
- 逻辑矛盾：逻辑矛盾可以通过以下方式发生：
 - 用户询问是否存在一个上位词，使得所有具有该上位词的词都被排除，而模型回答“是”。
 - 用户询问秘密单词是否有一个上位词，以至于所有剩余逻辑一致的单词都有那个上位词，而模型回答“没有”。

- 有一个逻辑上一致的单词剩下，用户猜测它，但模型说“不”。
- 用户猜一个已经被逻辑排除的词，而模型却回答“是”。
- 没有逻辑一致的词语剩下。
- 用户行动：用户模拟器不尝试通过最佳方式快速识别秘密词。相反，它基于概率状态机运行，有时会做出已经知道答案的猜测，试图在逻辑矛盾中抓住模型。每一轮，用户模拟器以给定的概率选择以下操作之一：
 - 即使仍然有多个逻辑上一致的词 ($p = 0.02$) 存在，也自发地尝试猜测那个秘密词。
 - 在猜测秘密单词时，选择一个已经在逻辑上排除的单词 ($p = 0.15$)。
 - 在猜测上位词时，应选择一个仅属于那些已被逻辑排除的词 ($p = 0.1$)。
 - 在猜测上位词时，选择一个属于所有剩余逻辑一致的单词的 ($p = 0.1$) 上位词。
 - 猜测一个上位词，该上位词属于非排除词的适当子集（否则）。
- 词汇集：在游戏中使用的词语及其相关的上位词是根据以下标准预先筛选的：
 - 这些词是出现在 SCOWL 词表中的名词。
 - 每个词至少有 7 个上位词。
 - 每个词都有上位词“物理实体”。
 - 没有词语具有“抽象”的上位词。
 - 没有两个词具有相同的上位词集合。
 - 在这个集合中，没有任何一个词是其他任何词的上位词。

A.3. 例子

Movie Recommendation

user

You're a movie recommendation agent who will interact with a user. Your goal is to ask questions in order to understand the user's preferences and suggest a movie that they'll enjoy the most. For this, you are given a library of movies in the table below. Each movie has scores on {num_attributes} different attributes: ...

The user's preference can be modeled as a linear function of the {num_attributes} scores and this function is hidden from you. The user will always act consistently based on the predefined user preference function.

...

You're only allowed to ask {num_questions} questions from the user, to uncover the underlying user preference function. Note that each question can only be formatted as follows:

"<question>Would you prefer watching \$option_1 over \$option_2?</question>".

...

The user will respond to your question. The response can only be "Yes", "No" or "No Preference". ...

Only once you're done interacting with the user after asking {num_questions} questions one-by-one, you will be given a new set of movies and based on whatever you've learnt about the user, your goal would be to pick the movie which you think the user will enjoy the most.

...

The movie dataset which you can use to ask the user questions is provided in the table below.

...

You can go ahead with asking the first question. While making deductions, try to uncover the underlying user preference function.

model

<question>Would you prefer watching Beneath the Crimson Tide over The Obsidian Fortress?</question>

user

Yes

Please ask the next question. 9 questions remaining. ...

...

user

No

Now apply whatever you learnt about the user and predict which movie will satisfy the user the most from the following set:

...

Reason about the linear user preference function and then provide your final answer. ...

model

<final_answer>A Dance with Time</final_answer>

Circuit Decoding

```

user
**Welcome to the Circuit Deduction Challenge!**

**Your Inventory:** You have a total of {components}. These are distributed
across {num_circuits} circuits, labeled {circuit_labels}.

**Your Goal:** Deduce the logical structure of each circuit.

**The Setup:**
* Each circuit accepts {num_inputs} binary inputs (0 or 1) and produces a single
  binary output (0 or 1).
* Circuits are implemented in layers. Each layer takes two inputs from the
  previous layer and produces a single output.
* No circuit will always output 0 or always output 1, regardless of input.
* All circuit components are being used and are distributed without replacement
  across {num_circuits} circuits.

**Your Tools:** You have {num_questions} queries to figure all the circuits out.
  Each query is a call to a circuit with specific inputs.
  At each turn of the conversation with me, you must make only 1 query. Hence,
  there will be {num_questions} turns of the conversation before you have to make
  your final guess.

**How to Play:**
1. At each turn, you must make two queries
2. Use the following format for your query:
  ...
3. You must make only 1 query at each turn

**Think Strategically:**
* Reason about the behavior of each circuit based on the available components.
* Plan your {num_questions} queries carefully to maximize information gained.

**Final Submission:**
After your {num_questions} queries, provide the truth tables for all circuits ({
circuit_labels}) in this format:
...

Please start with your first query. Plan your steps before making the queries.

model
<query>A(0, 0, 0)</query><ctrl100>

user
A(0, 0, 0) = 1
Please send your next query. 17 queries remaining. ...
...

user
C(0, 0, 0) = 0
Please provide the truth tables for all circuits in the format requested. ...

model
<guess>1011111101000101010111111</guess>

```

Twenty Questions

user

Let's play a game of 20 questions. I want you to choose a secret word (don't tell me what it is!) from the list below:

{word_list}

I will ask you simple yes/no questions about the word. You should answer these questions with either 'yes' or 'no', according to the attributes listed below:

{word_to_attributes_list}

So for example if your word is 'dog', and 'dog' has the listed attribute 'object', and I ask 'Is it a type of object?', you should say 'yes', even though dogs are not typically considered objects.

Conversely, if your word is 'dog', and 'dog' *does not* have the listed attribute 'pet', and I ask 'Is it a type of pet?', then you should say 'no', even though dogs are typically considered pets.

When I think I know what the word is, I'll ask you to confirm it, and you should again respond with either 'yes' or 'no'.

Ok, let me know once you've picked your word (remember, don't tell me what it is), and we'll get started!

model

Okay, I have chosen a secret word. Ask away!

user

Is it a type of plate?

model

no

...

Word Chaining

user

We're going to play a game. One of us will say a word, then the other will say a new word that starts with the same letter that the last word ended with. (So for example if I said 'ant', you might say 'television'.) All words must come from the list below. If you repeat a word, or select a word that isn't on the list, or break the first-letter rule, you lose. Please always wrap your chosen word in single-quote marks.

Allowed words: {words}

model

Okay, I'll start with 'encirclement'

user

Ok, my word is: 'topaz'.

...

Word Guess

user

Let's play word guess. The secret word is from the following vocabulary: {vocab }.

At each attempt, I will give feedback to you and specify which tiles are grey, yellow and green. Green implies that the letter is in the word and in the correct position, yellow implies that the letter is in the word but in the wrong position and grey implies that the letter is not in the word.

For e.g., if the secret word is SPARK and you guessed PROOF, then the feedback would be:

yellow, yellow, grey, grey, grey.

Your goal is to correctly guess the secret word in the minimum number of attempts and will be evaluated on the number of attempts you take to arrive at the secret word. Think step by step to ensure that you're taking the most optimal actions to guess the secret word.

All of your guesses should be structured as follows:

<attempt>Your guess</attempt>

Make sure that all of your attempts are strictly from the vocabulary specified above.

Start with your first attempt and make sure to follow the format specified above.

model

<attempt>RETRO</attempt>

user

green, grey, grey, grey, yellow

...