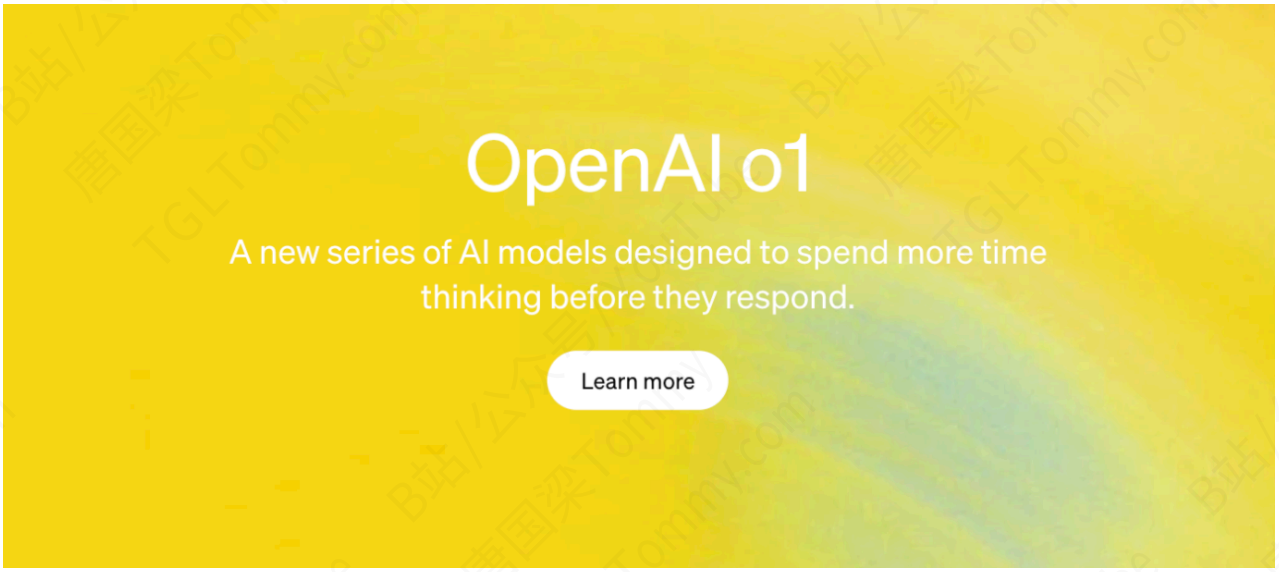
# 1. o1模型简介

## 1.1 版本发布

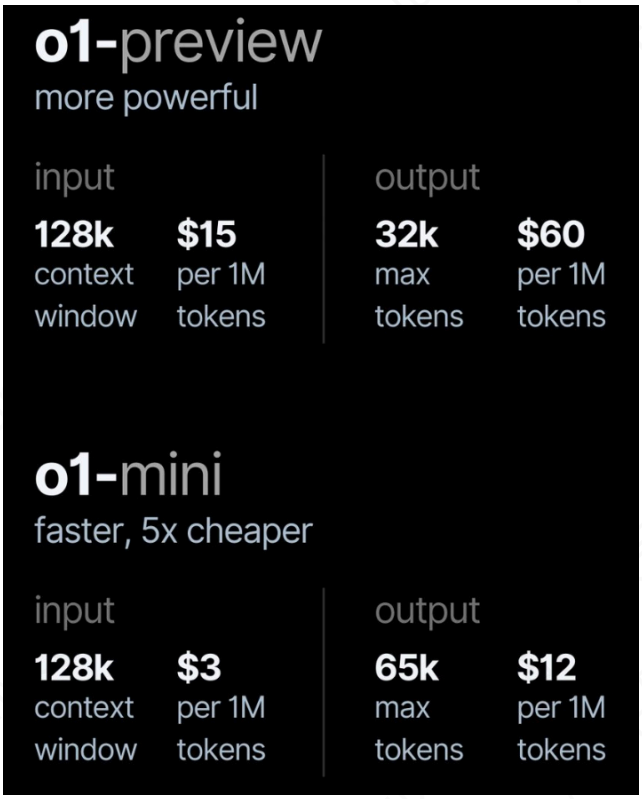
在2024年9月12日，OpenAI推出了一系列专为解决复杂问题设计的推理模型（“o1模型系列”），分别是：o1-preview（已上线），o1-mini（已上线），o1（未上线），这一系列模型能够处理比以前模型更复杂的任务，特别是在科学、编码和数学领域。该模型代表了人工智能能力的重大进步，并因此重新命名为OpenAI o1系列。

> ⭐趣闻与猜测⭐：OpenAI团队在开发者AMA中提到，o1是一个“模型”，而不是一个“系统”，“o1”中的字母“o”可能代表“猎户座（Orion）”。Sam Altman发布的推文提到他对冬季星座的期待，或许暗示了猎户座星座的隐喻。猎户座是北半球的冬季星座，可能象征OpenAI正在构建一个更大的AI系统，即“星座”，其中o1只是其中一颗星。OpenAI似乎在创建自己的AI模型“星座”，每个模型在广泛的、互联的系统中扮演不同的角色，协同工作，形成一种新型AI生态系统。

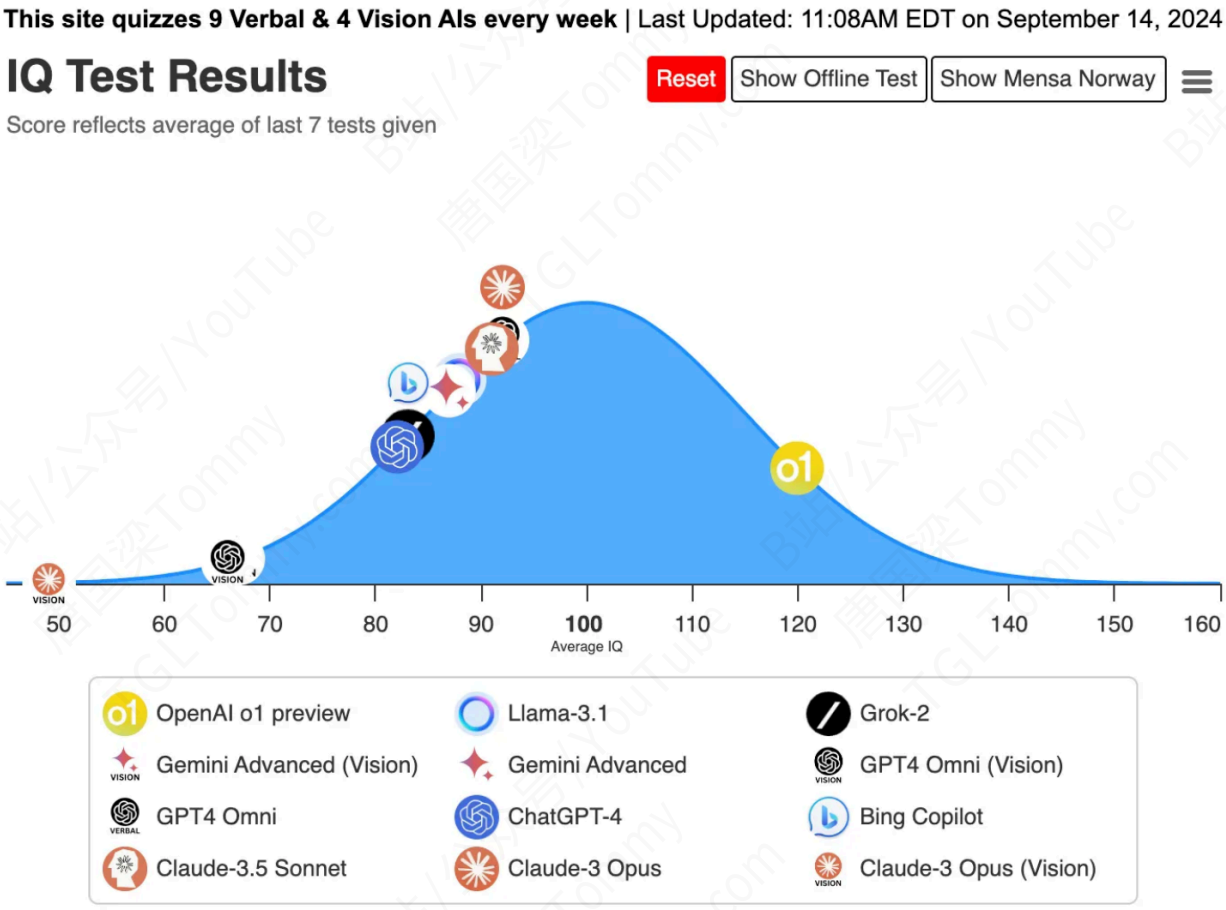



- **o1-preview**：这是o1模型的早期预览版，设计用于处理需要广泛通用知识的复杂问题。
- **o1-mini**：o1的一个更快速且更经济的版本，擅长于编码、数学和科学任务，尤其在不需要广泛通用知识的情况下表现出色，o1-mini比o1-preview便宜80%。

| Model Name | GPT-4o | o1-preview | GPT-4o mini | OpenAI o1-mini |
|---|---|---|---|---|
| Input Token Cost | $5.00 / 1M tokens | $15.00 / 1M tokens | $0.15 / 1M tokens | $3.00 / 1M tokens |
| Output Token Cost | $15.00 / 1M tokens | $60.00 / 1M tokens | $0.60 / 1M tokens | $12.00 / 1M tokens |

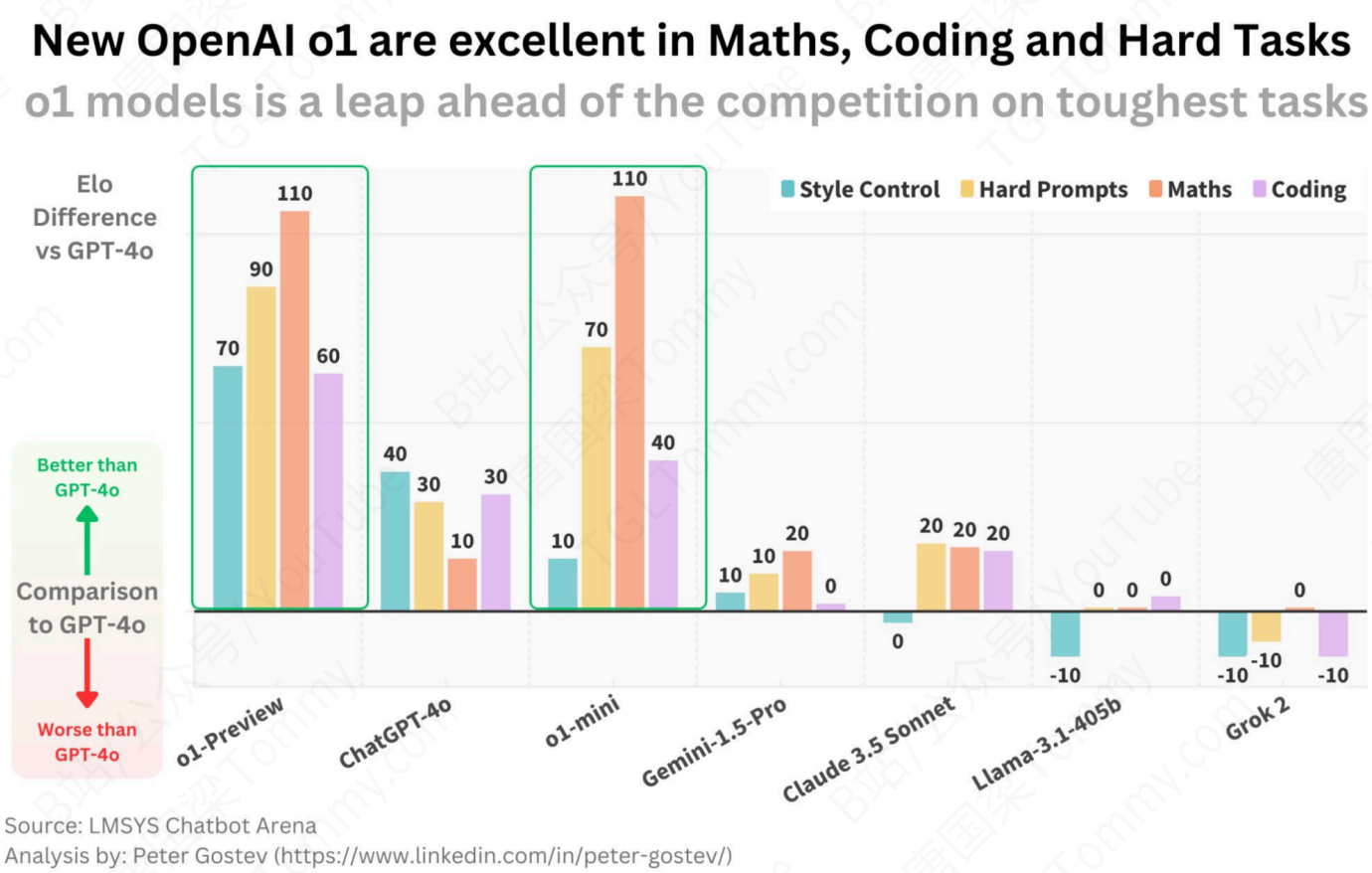



- **智商IQ测试：**

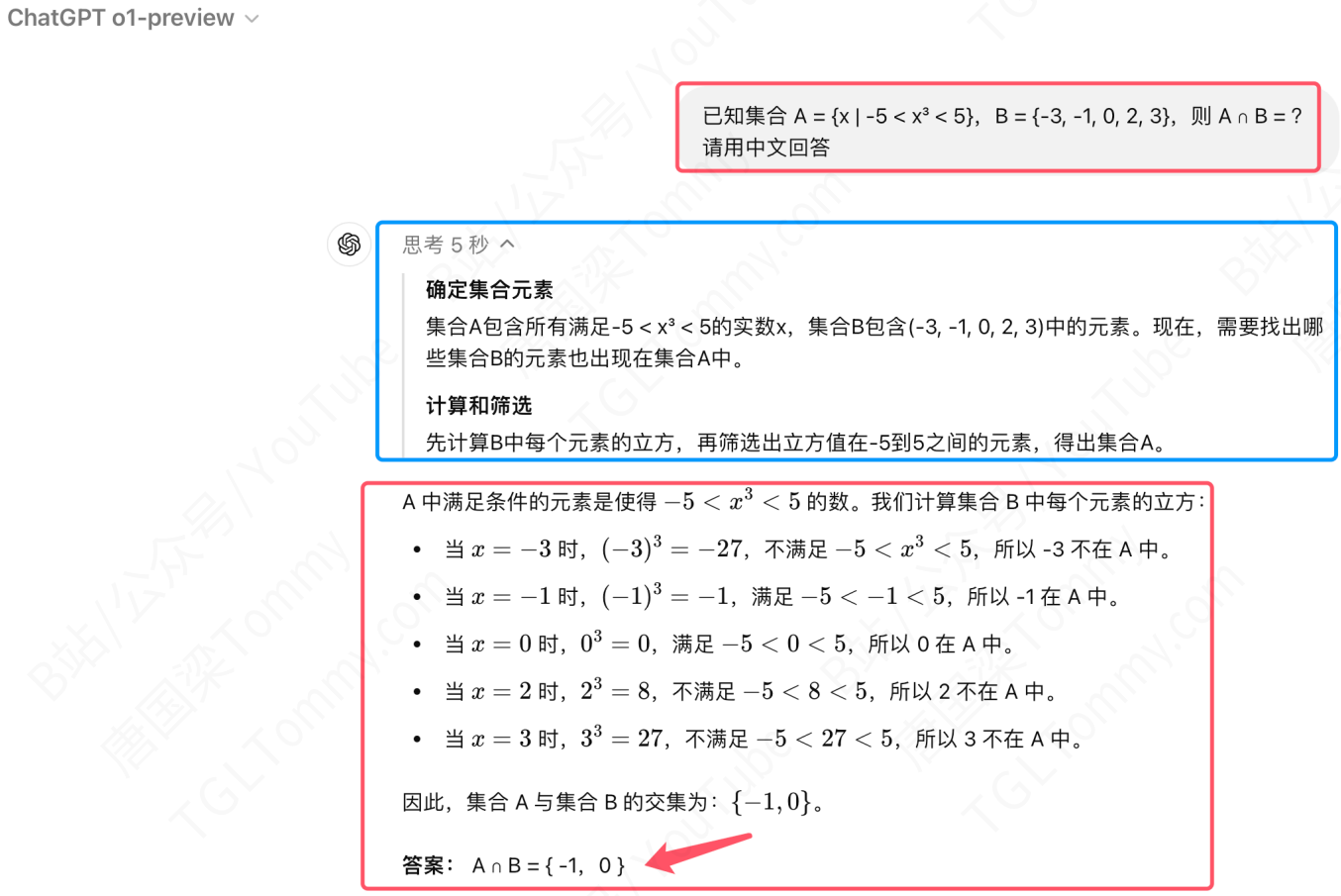



- **LMSYS Chatbot Arena 排名：**

| Rank* (UB) | Model | Arena Score | 95% CI | Votes | Organization | License | Knowledge Cutoff |
|---|---|---|---|---|---|---|---|
| 1 | o1-preview | 1355 | +12/-11 | 2991 | OpenAI | Proprietary | 2023/10 |
| 2 | ChatGPT-4o-latest (2024-09-03) | 1335 | +5/-6 | 10213 | OpenAI | Proprietary | 2023/10 |
| 2 | o1-mini | 1324 | +12/-9 | 3009 | OpenAI | Proprietary | 2023/10 |
| 4 | Gemini-1.5-Pro-Exp-0827 | 1299 | +5/-4 | 28229 | Google | Proprietary | 2023/11 |
| 4 | Grok-2-08-13 | 1294 | +4/-4 | 23999 | xAI | Proprietary | 2024/3 |
| 6 | GPT-4o-2024-05-13 | 1285 | +3/-3 | 90695 | OpenAI | Proprietary | 2023/10 |
| 7 | GPT-4o-mini-2024-07-18 | 1273 | +3/-3 | 30434 | OpenAI | Proprietary | 2023/10 |
| 7 | Claude 3.5 Sonnet | 1269 | +3/-3 | 62977 | Anthropic | Proprietary | 2024/4 |
| 7 | Gemini-1.5-Flash-Exp-0827 | 1269 | +4/-4 | 22264 | Google | Proprietary | 2023/11 |
| 7 | Grok-2-Mini-08-13 | 1267 | +4/-5 | 22041 | xAI | Proprietary | 2024/3 |
| 7 | Gemini Advanced App (2024-05-14) | 1267 | +3/-3 | 52218 | Google | Proprietary | Online |
| 7 | Meta-Llama-3.1-405b-Instruct-fp8 | 1266 | +4/-4 | 31280 | Meta | Llama 3.1 Community | 2023/12 |




- **示例：2024 高考数学-选择题**：




## 1.2 模型特点

OpenAI的o1模型通过模仿人类的“思维链”来解决问题。借助强化学习，模型能够不断优化其思维链，识别并纠正错误，将复杂问题分解为更简单的步骤进行处理。

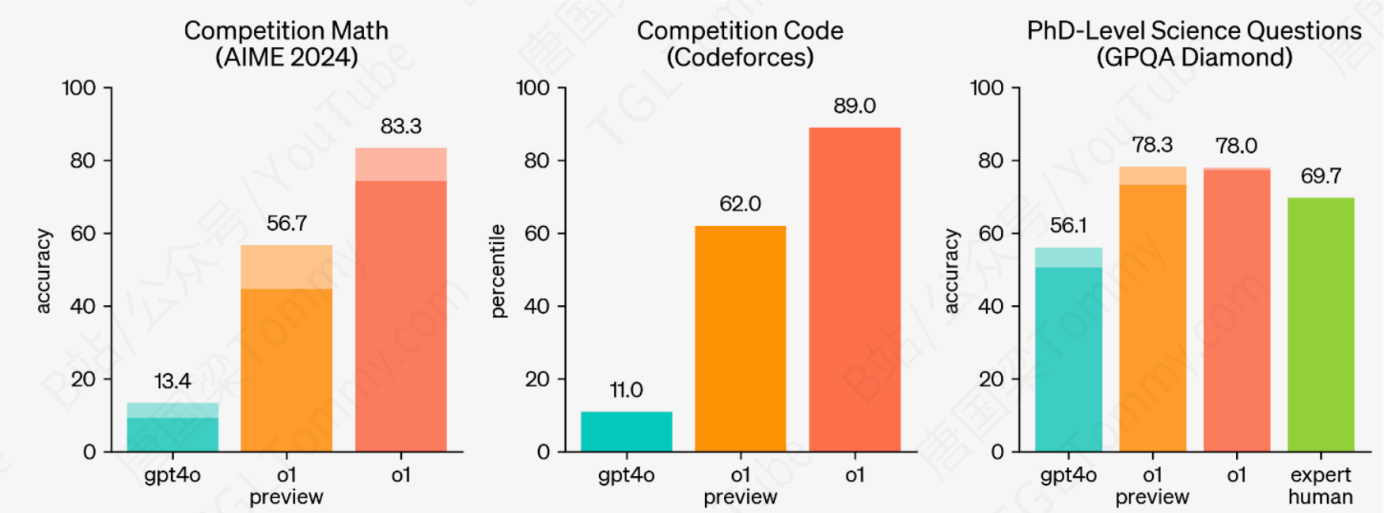
- **训练**：o1模型采用高效的强化学习算法，在训练过程中以较少的数据获得显著的效果。随着训练时间的增加，模型性能也随之持续提升。
- **推理**：o1模型的独特之处在于其在回答前会“思考”。它将大问题分解为多个小步骤，并在每个步骤中进行判断，识别正确与错误。这种“多步骤推理”虽然并非全新概念，但由于过去的技术限制，难以实现。而o1模型通过这种方式，有效地帮助用户逐步解决复杂问题，处理大型任务更加高效。

> 关键技术
>
> ① **强化学习**：模型通过探索不同策略、识别错误并调整其方法，来不断完善推理过程，从而得出最准确的解决方案。
>
> ② **思维链推理**：这是一种将复杂问题分解为更小的可控组件的技术，类似于在回答问题前先思考所有步骤，确保推理过程完整。
>
> ③ **推理时间扩展**：在推理过程中动态调整计算量，随着任务复杂度的增加，模型花费更多时间进行推理。

# 2. o1模型性能

## 2.1 竞赛表现

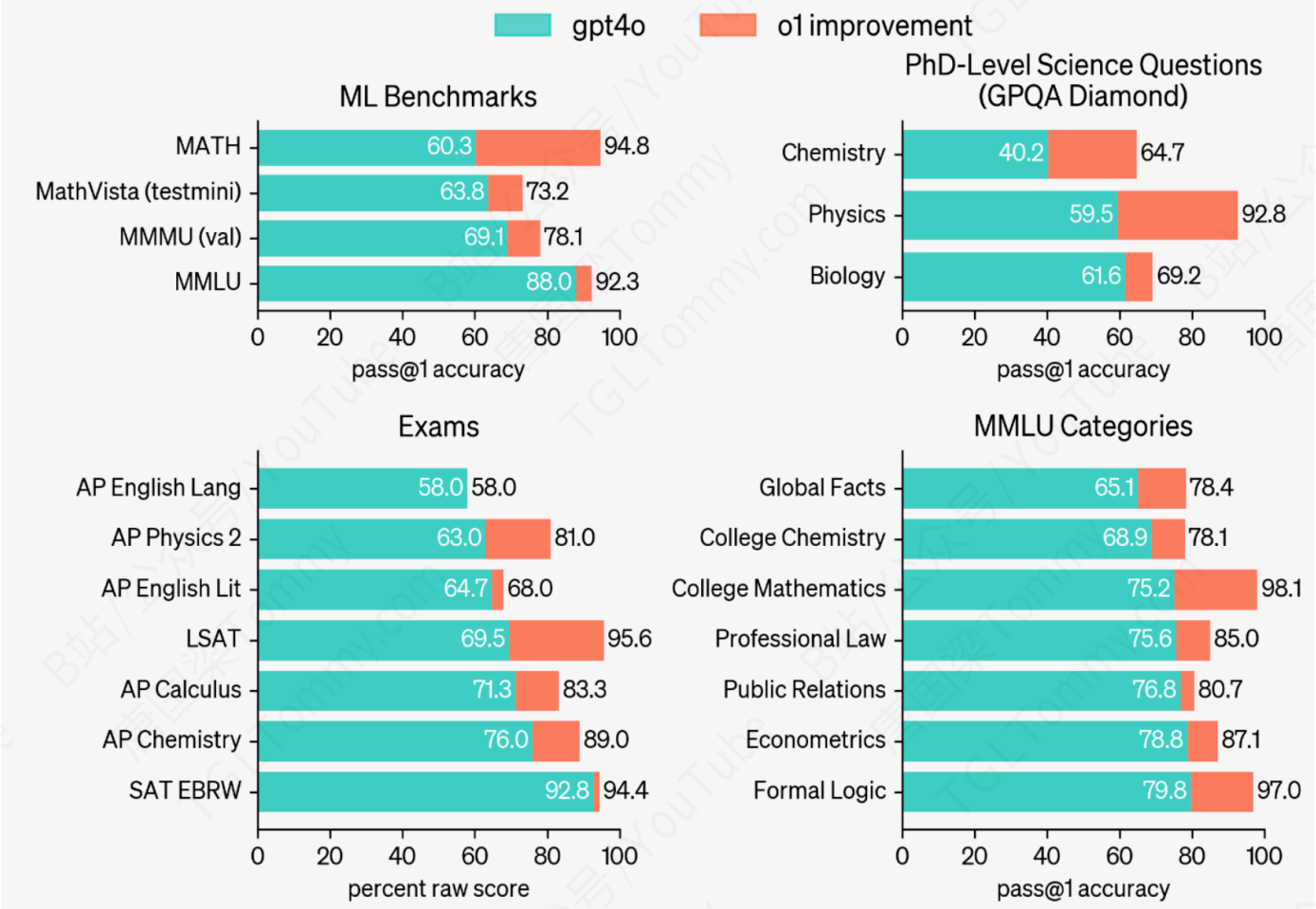
- 在编程竞赛问题（Codeforces）中，该o1模型达到了前89%的水平；
- 在美国数学奥林匹克（AIME）预选赛中进入了美国前500名，，GPT-4o仅正确解决了13%的问题，而o1模型达到了83%的正确率；
- 在一个复杂的物理、化学、生物学智能基准测试中，o1的表现超越了拥有博士学位的人类专家。




o1 greatly improves over GPT-4o on challenging reasoning benchmarks. Solid bars show pass@1 accuracy and the shaded region shows the performance of majority vote (consensus) with 64 samples.

## 2.2 基准测试

- **评估数据**: 针对人类考试和机器学习基准测试（ML benchmarks），在多数推理密集型任务中，o1显著优于GPT-4。
- **MMLU表现**: 在视觉感知能力启用的情况下，o1在MMLU的57个子类别中击败GPT-4o，达到54个的领先表现，且整体得分为78.2%。

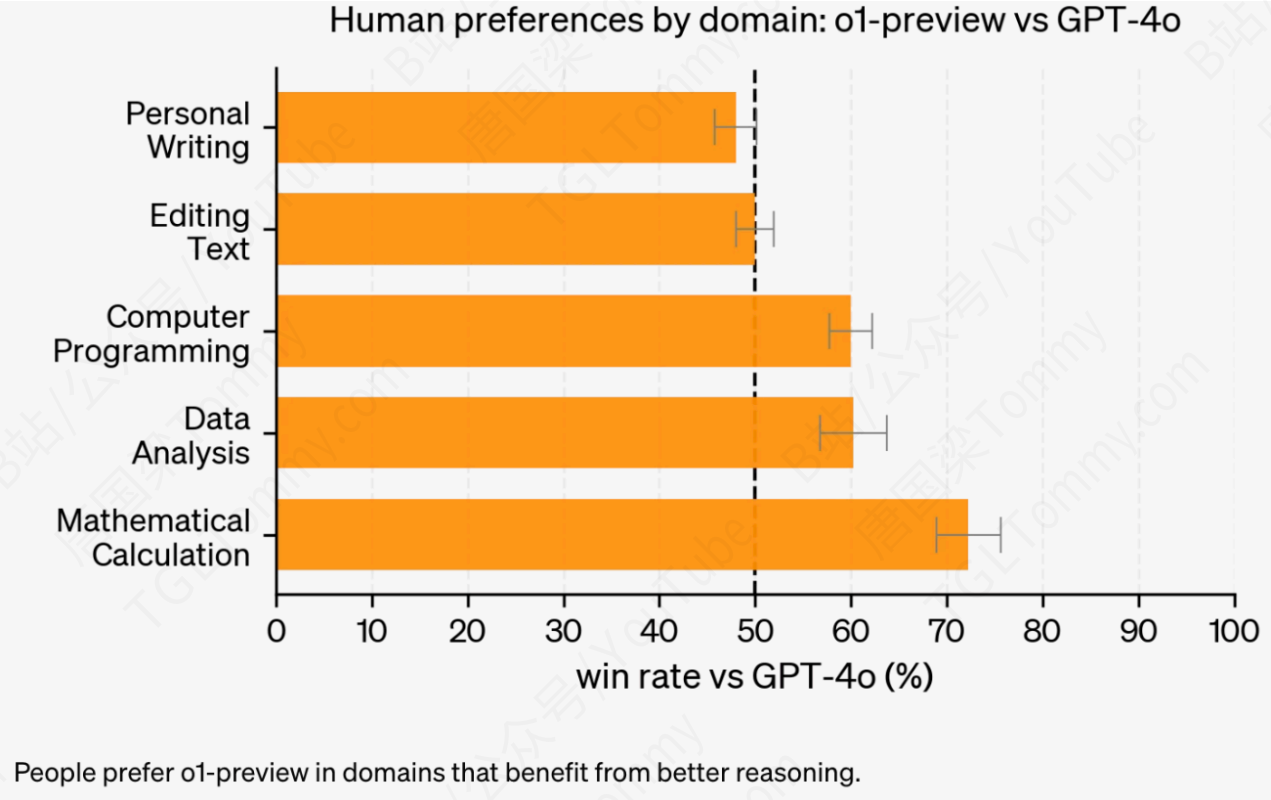



o1 improves over GPT-4o on a wide range of benchmarks, including 54/57 MMLU subcategories. Seven are shown for illustration.

Table 12: MMLU Language (0-shot)

| Language | o1-preview | GPT-4o | o1-mini | GPT-4o-mini |
|---|---|---|---|---|
| Arabic | **0.8821** | 0.8155 | 0.7945 | 0.7089 |
| Bengali | **0.8622** | 0.8007 | 0.7725 | 0.6577 |
| Chinese (Simplified) | **0.8800** | 0.8335 | 0.8180 | 0.7305 |
| English (not translated) | **0.9080** | 0.8870 | 0.8520 | 0.8200 |
| French | **0.8861** | 0.8437 | 0.8212 | 0.7659 |
| German | **0.8573** | 0.8292 | 0.8122 | 0.7431 |
| Hindi | **0.8782** | 0.8061 | 0.7887 | 0.6916 |
| Indonesian | **0.8821** | 0.8344 | 0.8174 | 0.7452 |
| Italian | **0.8872** | 0.8435 | 0.8222 | 0.7640 |
| Japanese | **0.8788** | 0.8287 | 0.8129 | 0.7255 |
| Korean | **0.8815** | 0.8262 | 0.8020 | 0.7203 |
| Portuguese (Brazil) | **0.8859** | 0.8427 | 0.8243 | 0.7677 |
| Spanish | **0.8893** | 0.8493 | 0.8303 | 0.7737 |
| Swahili | **0.8479** | 0.7708 | 0.7015 | 0.6191 |
| Yoruba | **0.7373** | 0.6195 | 0.5807 | 0.4583 |

- **人类偏好评估**：在多个领域的复杂、开放式提示上对o1-preview与GPT-4o进行了对比评估。人类训练师更偏爱o1-preview，尤其是在推理密集型任务（如数据分析、编码和数学）中，o1-preview表现优于GPT-4o。o1-preview在某些自然语言任务中的表现不如GPT-4o，表明其并不适合所有的使用场景。




People prefer o1-preview in domains that benefit from better reasoning.

- **安全评估**：思维链推理为模型的对齐和安全提供了新的机会。通过将模型行为的策略融入推理模型的思维链中，能更有效地教授模型人类的价值观和原则。

| 指标 | GPT-4o | o1-preview |
|---|---|---|
| **% 有害提示的安全完成率** 标准 | 0.990 | 0.995 |
| **% 有害提示的安全完成率** 具有挑战性的：越狱和边缘案例 | 0.714 | 0.934 |
| ↳ 暴力或犯罪骚扰（一般） | 0.845 | 0.900 |
| ↳ 非法性内容 | 0.483 | 0.949 |
| ↳ 涉及未成年人的非法性内容 | 0.707 | 0.931 |
| ↳ 针对受保护群体的暴力或犯罪骚扰 | 0.727 | 0.909 |
| ↳ 非暴力不法行为的建议 | 0.688 | 0.961 |
| ↳ 暴力不法行为的建议 | 0.778 | 0.963 |
| ↳ 自残的建议或鼓励 | 0.769 | 0.923 |
| **% WildChat 中每类最高 Moderation API 分数的前 200 条的安全完成率** | 0.945 | 0.971 |
| **Goodness@0.01 StrongREJECT 越狱评估** | 0.220 | 0.840 |
| **人源越狱评估** | 0.770 | 0.960 |
| **% 内部良性边缘案例的合规性** “ 非过度拒绝” | 0.910 | 0.930 |
| **% XSTest 中良性边缘案例的合规性**“ 非过度拒绝” | 0.924 | 0.976 |

- **综合评估结果**

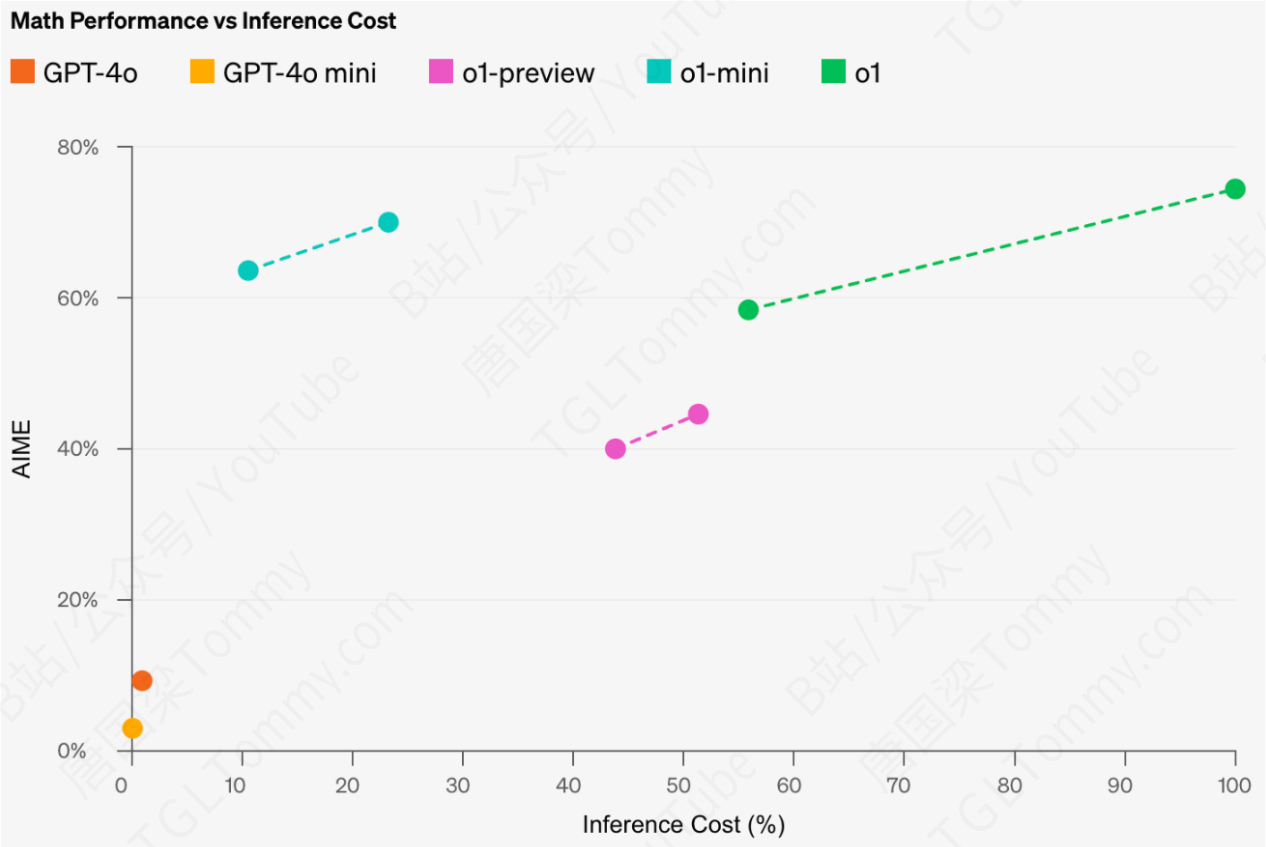
| Dataset | Metric | gpt-4o | o1-preview | o1 |
|---|---|---|---|---|
| **Competition Math** AIME (2024) | cons@64 | 13.4 | 56.7 | 83.3 |
| | pass@1 | 9.3 | 44.6 | 74.4 |
| **Competition Code** CodeForces | Elo | 808 | 1,258 | 1,673 |
| | Percentile | 11.0 | 62.0 | 89.0 |
| **GPQA Diamond** | cons@64 | 56.1 | 78.3 | 78.0 |
| | pass@1 | 50.6 | 73.3 | 77.3 |
| **Biology** | cons@64 | 63.2 | 73.7 | 68.4 |
| | pass@1 | 61.6 | 65.9 | 69.2 |
| **Chemistry** | cons@64 | 43.0 | 60.2 | 65.6 |
| | pass@1 | 40.2 | 59.9 | 64.7 |
| **Physics** | cons@64 | 68.6 | 89.5 | 94.2 |
| | pass@1 | 59.5 | 89.4 | 92.8 |
| **MATH** | pass@1 | 60.3 | 85.5 | 94.8 |
| **MMLU** | pass@1 | 88.0 | 90.8 | 92.3 |
| **MMMU (val)** | pass@1 | 69.1 | n/a | 78.1 |
| **MathVista (testmini)** | pass@1 | 63.8 | n/a | 73.2 |

# 3. OpenAI o1-mini

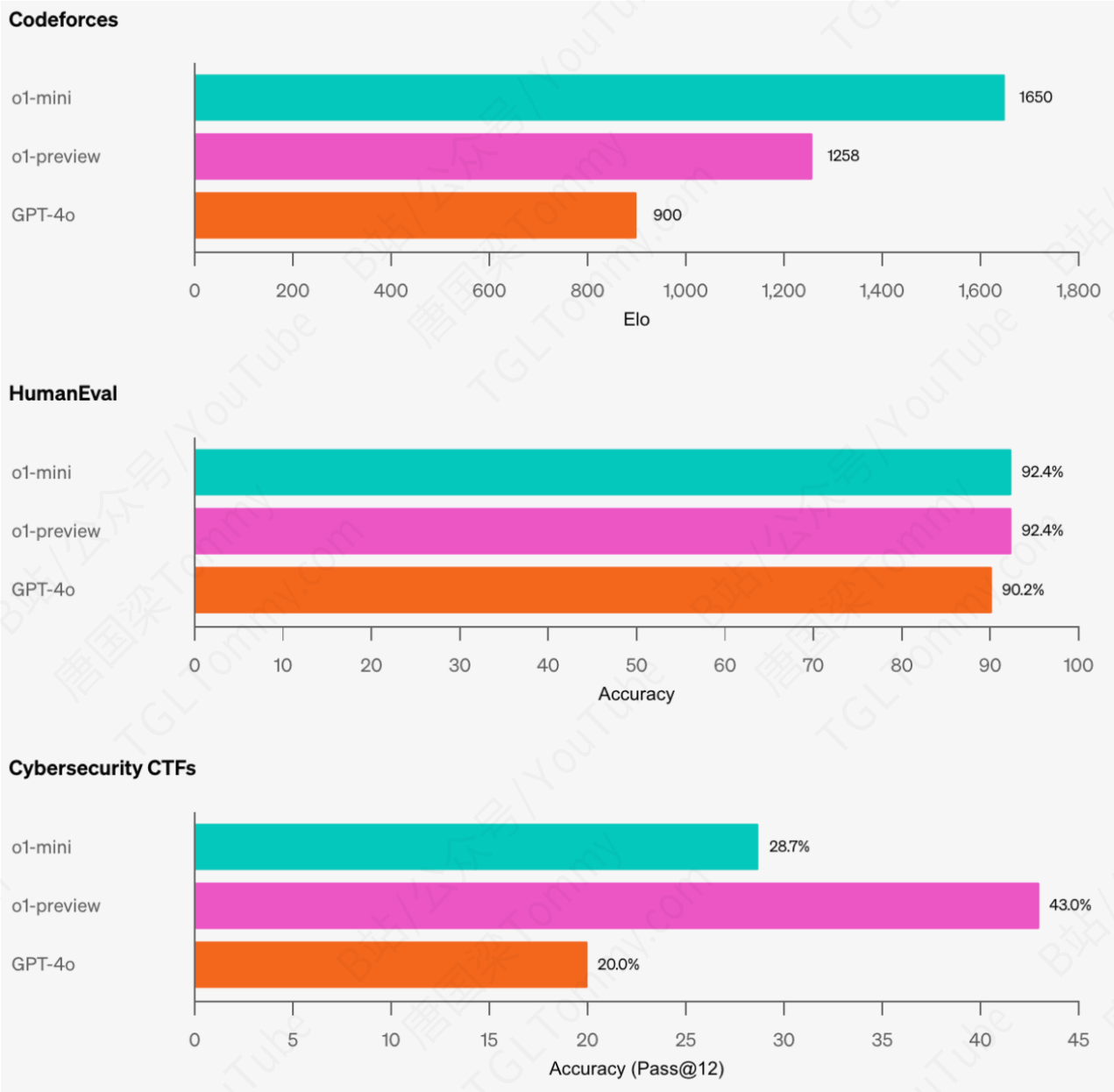
**特点**：o1-mini是一个成本高效的推理模型，特别在 STEM（科学、技术、工程、数学）领域表现优异，尤其是数学和编程。该模型的性能在多个评估基准上（如AIME和Codeforces）几乎与OpenAI o1相当。相比o1-preview，o1-mini提供更高的速率限制和更低的延迟。

## 3.1 数学和编程表现

**数学**：在高中AIME数学竞赛中，o1-mini的表现（70.0%）接近o1（74.4%），并且远超o1-preview（44.6%）。o1-mini的分数（约 11/15 题）将其排名在美国高中生前500名左右。

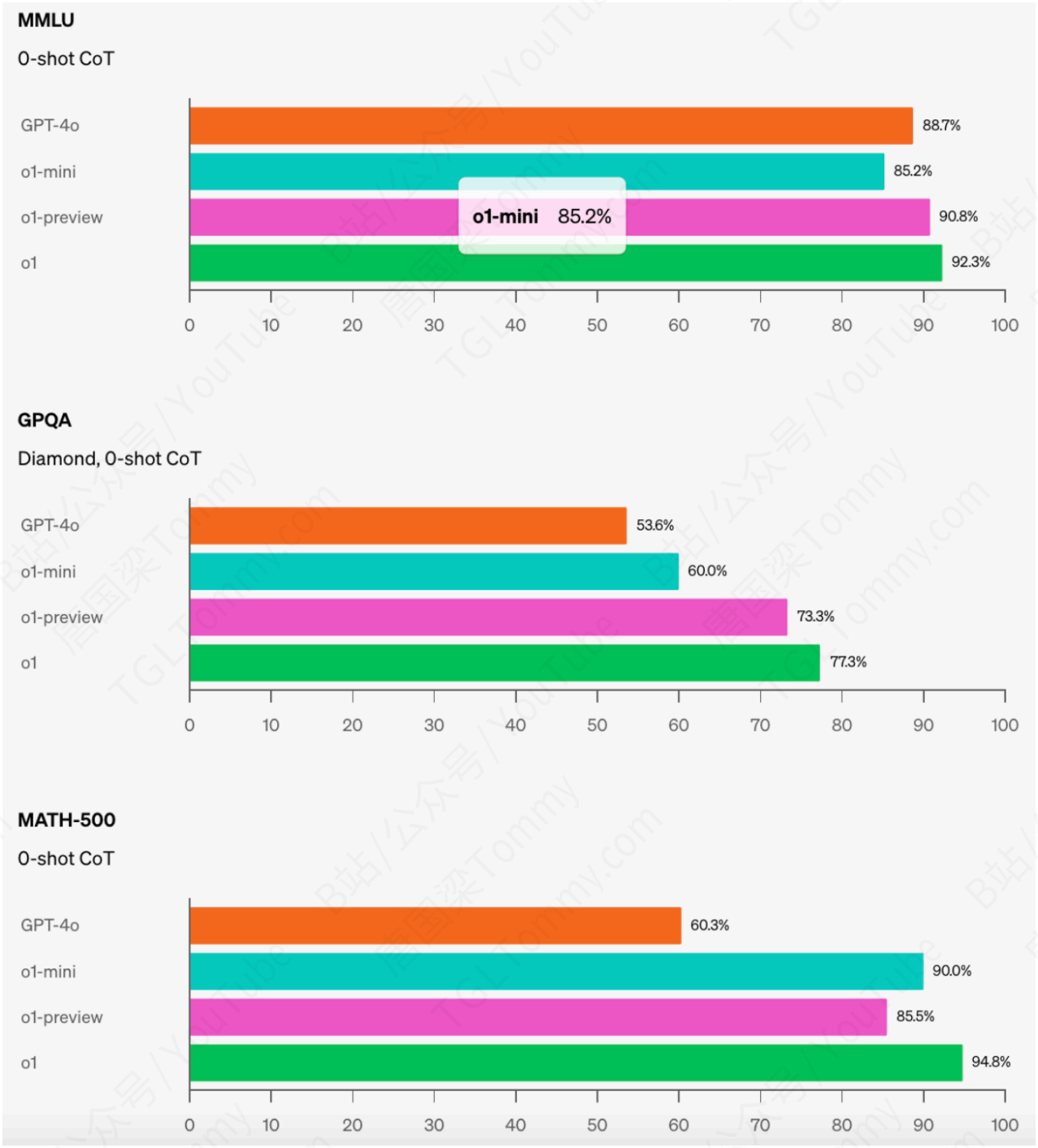



**编程**：在Codeforces编程竞赛中，o1-mini达到了1650 Elo分数，接近o1（1673），并且明显高于o1-preview（1258）。这一Elo分数使其处于Codeforces平台上编程者的前86%水平。此外，o1-mini在HumanEval编程基准测试和高中级别的网络安全夺旗赛（CTF）中的表现也很出色。



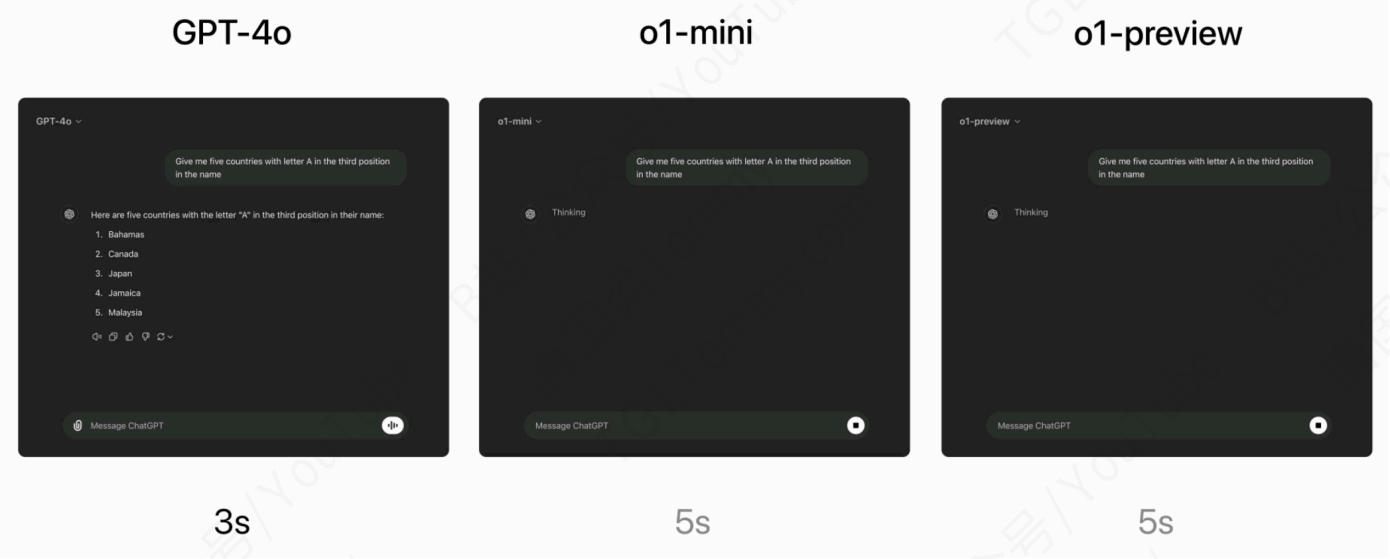

## 3.2 STEM领域的表现

在一些学术基准测试中（如GPQA科学任务和MATH-500），o1-mini的推理能力超过了GPT-4o。但在MMLU等任务上，o1-mini的表现不如GPT-4o，在GPQA基准测试中也落后于o1-preview，主要因为缺乏广泛的世界知识。




## 3.3 模型速度

在一个词语推理问题上，比较了GPT-4o、o1-mini和o1-preview的响应速度。GPT-4o未能正确回答，而o1-mini和o1-preview都给出了正确答案，其中o1-mini达到答案的速度比GPT-4o快了3-5 倍。




## 3.4 安全评估

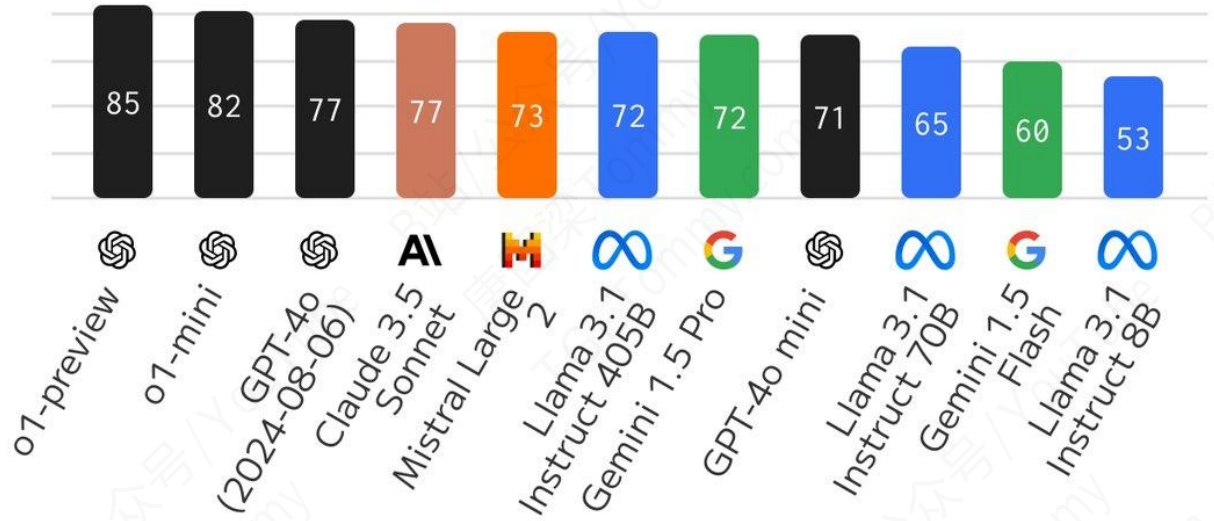
**训练与评估**：o1-mini使用与o1-preview相同的对齐和安全技术进行训练。该模型在内部版本的StrongREJECT数据集上的“越狱”鲁棒性比 GPT-4o高59%。

| 指标 | GPT-4o | o1-mini |
| --- | --- | --- |
| **% 有害提示的安全拒绝完成率** (标准) | 0.99 | 0.99 |
| **% 有害提示的安全完成率** (具有挑战性的：越狱和边缘案例) | 0.714 | 0.932 |
| **% 良性边缘案例的合规性** (“非过度拒绝”) | 0.91 | 0.923 |
| **Goodness@0.1 StrongREJECT 越狱评估** (Souly et al. 2024) | 0.22 | 0.83 |
| **人源越狱评估** | 0.77 | 0.95 |

# 4. o1模型的优势与局限

- **优势**
  - 在需要深度推理的任务中，o1能够提供详尽的分步解答，有助于解决复杂问题。
- **局限**
  - 在处理简单问题时，o1往往会过度思考，导致回答冗长且不必要。因此，它在日常简单任务中的实用性较低。
  - 相较于其他模型，o1的独特之处在于其价格高昂。通常，模型的费用是基于输入和输出的token数量，但o1增加了额外的“推理 token”，这些token是模型分解大问题时的隐藏计算步骤。虽然用户无法直接看到这些计算，但仍会为它们付费，因此使用o1时需要特别小心，避免在处理简单问题时过多消耗token。

*Artificial Analysis Quality Index; Higher is better*




## SPEED

*Output Tokens per Second; Higher is better*

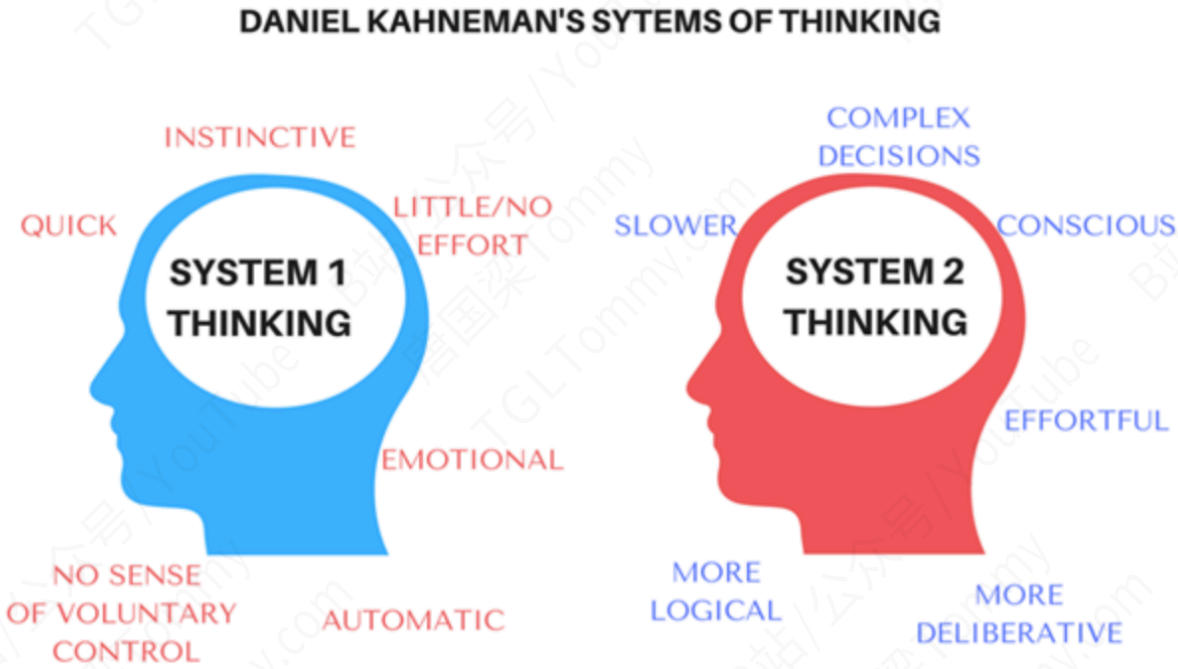

## PRICE

*USD per 1M Tokens; Lower is better*


# 5. OpenAI o1模型系列 VS GPT

传统的大型语言模型（如GPT）主要在三个阶段中使用计算资源：**预训练、微调和推理**。GPT-4 在预训练阶段消耗了大量的计算资源，尤其在其参数规模逐年增长的情况下，预训练的大规模数据处理让模型“更聪明”。

- 在推理阶段，尤其当大量用户使用模型时，资源的分配变得更加复杂，为简单问题和复杂问题使用相同的计算资源并不经济。
- OpenAI o1模型采用了新的推理范式。通过强化学习机制，o1学会在推理时合理分配资源，处理复杂查询时花费更多资源，类似于人类的**系统2思维**，即慢速、深思熟虑的推理方式。相比之下，GPT 更倾向于使用简单、快速的**系统1思维**。




# 6. o1-preview与o1-mini演示

## 6.1 推理工作原理

- **推理tokens**：o1 系列模型引入了“推理tokens”。模型使用这些tokens进行“思考”，即在生成可见的完成tokens之前，模型通过推理tokens分解对提示的理解，考虑多个生成响应的途径。在生成完成tokens后，推理tokens会被丢弃。

**示例：**

```
oyfjdnisdr rtqwainr acxz mynzbhhx -> Think step by step

请用上面的示例进行解码：

oyekaijzdf aaptcg suaokybhai ouow aqht mynznvaatzacdfoulxxz
```

**图-1：** 推理后的答案 ✅

oyfjdnisdr rtqwainr acxz mynzbhhx -> Think step by step

请用上面的示例进行解码:

oyekaijzdf aaptcg suaokybhai ouow aqht mynznvaatzacdfoulxxz

思考 53 秒

根据给定的解码示例，我们可以推断出每对加密字母对应一个解码字母，具体方法是将每对字母的字母序号相加，然后除以2，得到的平均值对应解码字母的字母序号。

以下是详细的解码步骤：

1. **将加密文本分成字母对**：

bash 复制代码

```
oy ek ai jz df aa pt cg su ao ky bh ai ou ow aq ht my nz nv aa tz ac df ou lx
```

2. **计算每对字母的平均值并找到对应的字母**：
   - `o(15) + y(25) = 40; 40 / 2 = 20; 20 -> T`
   - `e(5) + k(11) = 16; 16 / 2 = 8; 8 -> H`
   - `a(1) + i(9) = 10; 10 / 2 = 5; 5 -> E`
   - `j(10) + z(26) = 36; 36 / 2 = 18; 18 -> R`
   - `d(4) + f(6) = 10; 10 / 2 = 5; 5 -> E`
   - `a(1) + a(1) = 2; 2 / 2 = 1; 1 -> A`
   - `p(16) + t(20) = 36; 36 / 2 = 18; 18 -> R`
   - `c(3) + g(7) = 10; 10 / 2 = 5; 5 -> E`
   - `...`
3. **将解码后的字母组合成消息**：

mathematica 复制代码

```
THERE ARE THREE RS IN STRAWBERRY
```

**最终解码结果：**

**“THERE ARE THREE RS IN STRAWBERRY”**

**图-2：**推理过程如下

思考 53 秒

**解码过程和指导**

我正在解码新文本时，基于用户给定的示例进行操作。

**对照文本**

我正在对照"oyfjdnisdr rtqwainr acxz mynzbhhx"和"Think step by step"，发现长度不一致。需要更细致地调整或重新评估映射策略。

**识别字符映射**

正在分析字符映射，观察可能的字符替换方法，逐步找出解码的可能性。

解码信息

**重新排列字母**

正在尝试重排"oyfjdnisdr"中的字母，看看是否能得到"Think"。这让我想到，压缩长文本可能很有帮助。

**映射字母对**

我正在破解加密单词的字母对映射到解码单词的单字母映射。已经将字母对映射到解码字母，接着继续检查其他单词的映射。

Unlocked: 我尝试将每个字符对映射到字母，可能涉及ASCII码、字母位置和字符对转换等方法。

**分析字母位置**

我正在查看字母o和y在字母表中的位置，分别是15和25。试图找到如何从这两个位置得到T。

**探索不同方法**

通过字母编码计算组合，试图解码目标字母。有些计算得出结果K、N、H和I，显示出潜在的规律。

**解编码**

我正在尝试用字母对的中间字母解码，通过计算字母的位置平均值来解码。验算了多个例子，结果都正确。

**提取每对字母**

我从消息中提取每对字母，计算它们在字母表中的位置，通过平均得到目标字母的编码位置。这样我就能解码消息中的每个字母。

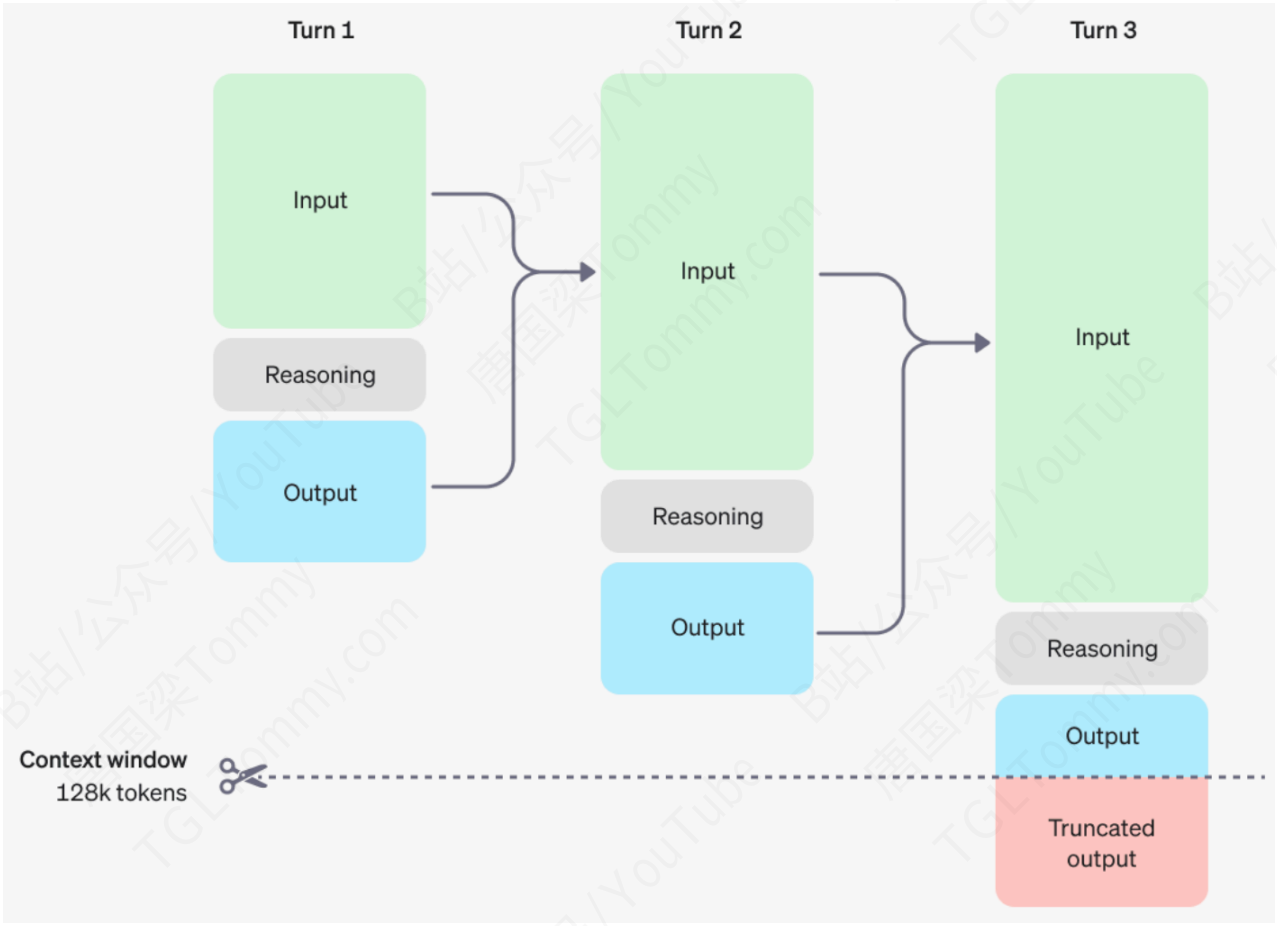
**对中英文混合文本的处理**

我正在整理这个消息，确保正确统计和处理其中的英文和中文字符。

> **朗读信笺**
> 我正在将编码后的字母分成对，共查出28对。字母总数为56个。
>
> **检查编码**
> 我在分析编码消息，发现最后一个字母与前24对匹配，但总字母数为56，而只有27对。这让我再数了一遍，结果与预期一致。
>
> **核对字母对**
> 我正在核对字母对，最后一个字母z没有配对。可能是之前统计出了问题。希望能找到其他可能的错误。
>
> **解码加密消息**
> 我正在计数字母，花时间为每个字母上数字。这让我能够识别出每个字母的总数。
>
> **分析字母数量**
> 我正在将54个字母分成27对，确保每对都包含两个字母。通过这种方式，我可以有效地对字母进行分类。
>
> 对这些配对进行解码
>
> **匹配内容**
> 我正在计算每对字母的平均值，并将其转换为相应的字母。这样，我终于得到了一个词组："THEREARE"。
>
> **解码信号**
> 我正在解码信号对组合的字母。结果是"THEREARETHREERS"，这引发了我对可能的计算错误的怀疑。
>
> **解码字母**
> 我正在检查每个字母对的平均值，并查找相应的字母。比如，Pair11的平均值是18，对应的字母是R。
>
> **解析字符**
> 通过字符对的平均值，将其转化为字母。最终，字母序列形成"THERE ARE THREE ERRORS"，这反映了一个明显的错误。

# 6.2 上下文窗口管理

- o1-preview 和 o1-mini 模型的上下文窗口限制为 128,000 个tokens。每个完成请求都有输出tokens的上限，包括不可见的推理tokens和可见的完成tokens：
  - o1-preview 的输出tokens上限为 32,768 个tokens。
  - o1-mini 的输出tokens上限为 65,536 个tokens。



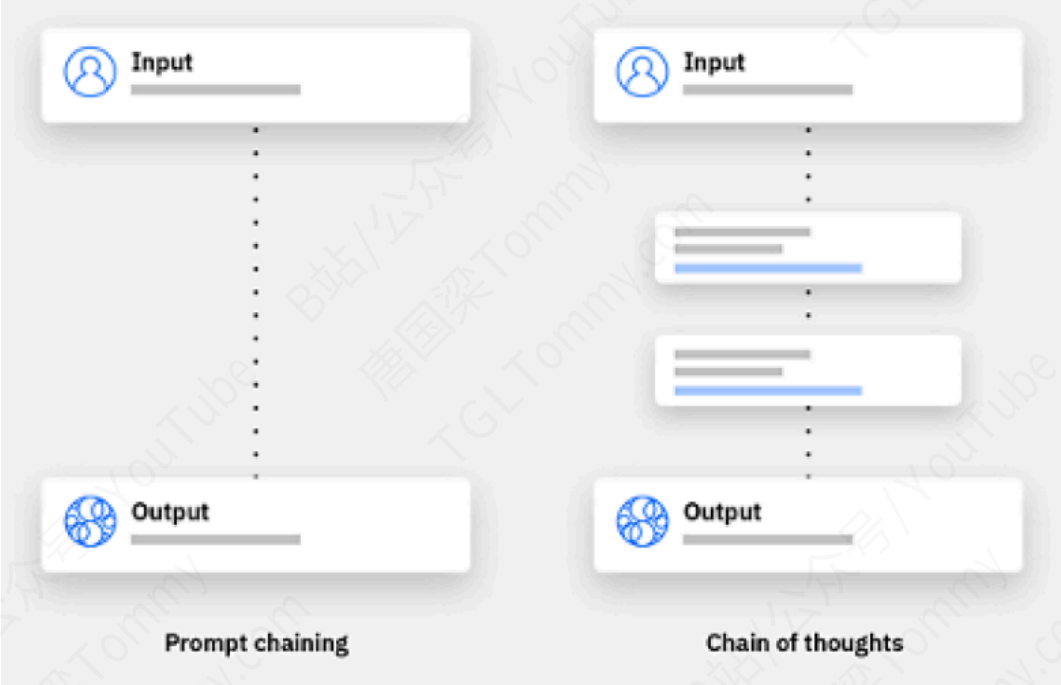

## 6.3 提示设计建议

- **简明直接**：保持提示简洁，避免复杂的指令。
- **避免思维链提示**：模型已内置推理能力，无需明确要求“逐步思考”或“解释推理”。
- **使用分隔符**：使用清晰的符号（如引号、XML 标签）帮助模型区分不同部分的输入。
- **限制上下文信息**：在检索增强生成中，仅提供最相关的信息，避免让模型处理过多上下文。

# 7. CoT（Chain-of-Thought）

## 7.1 工作原理

CoT提示技术通过**显式引导模型逐步生成解释或推理过程**，从而提高语言模型的性能，避免推理过程中跳过中间步骤。这种方法帮助模型逐步分解问题，并确保在到达最终答案之前不会遗漏关键的推理步骤。



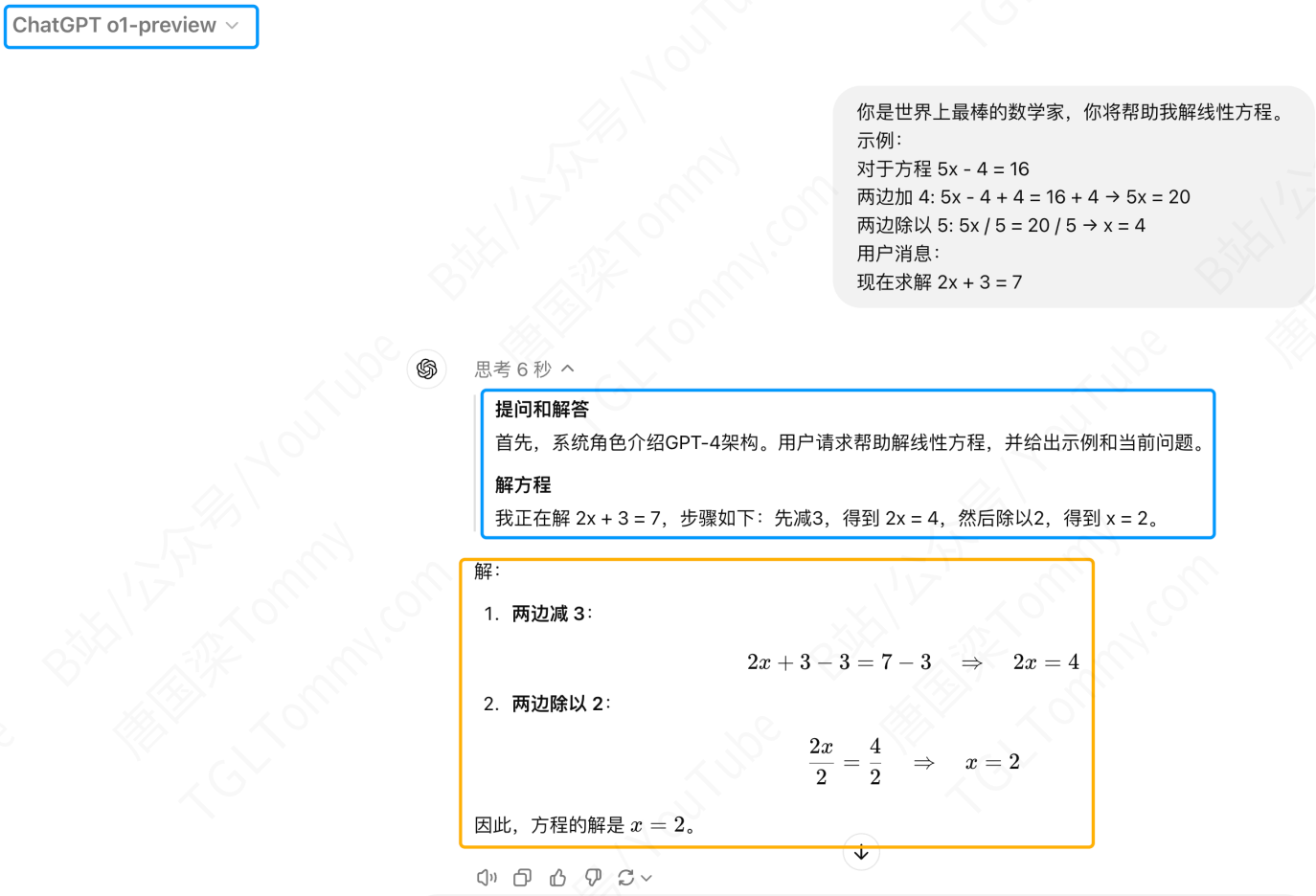

CoT提示的有效性在于它能**集中模型的注意力**，通过逐步分解推理任务，使模型专注于一次解决问题的一部分，从而**减少因处理过多信息导致的错误**。

可以通过以下策略进行：

- **显式指令**：通过明确的指令将问题分解为具体的步骤。例如，使用“首先，我们需要考虑……”这样的句子来引导模型详细思考每个步骤。
- **隐式指令**：有时不需要在提示中自己分解问题。只需在提示末尾添加“让我们一步一步思考”，即可促使模型推理并完成任务。

**示例：**

```
你是世界上最棒的数学家，你将帮助我解线性方程。
示例：
对于方程 5x - 4 = 16
两边加 4：5x - 4 + 4 = 16 + 4 → 5x = 20
两边除以 5：5x / 5 = 20 / 5 → x = 4
用户消息：
现在求解 2x + 3 = 7
ChatGPT
```




# 7.2 如何在提示中应用CoT？

- **结构化模板**：使用显式说明步骤的模板来引导模型完成任务。
- **交互式提示**：通过与模型进行对话，逐步解释每个步骤，或者提供反馈和修正。
- **反馈循环**：加入反馈机制，评估和修正模型的中间步骤，以确保正确性。

# 7.3 CoT优势与限制

## 7.3.1 优势

- **提高准确性**：将问题分解为较小的可管理步骤，帮助模型更准确地处理复杂任务。
- **增强可解释性**：逐步推理过程更加透明，使用户能够理解模型的最终结论。

## 7.3.2 限制

- **模型依赖性**：CoT是一种依赖模型能力的技术，其有效性取决于模型的底层能力。
- **提示生成**：有效的CoT提示需要精心设计，更新和维护不同任务类型的提示可能耗时且需要不断优化。

- **性能问题**：对于没有清晰顺序推理过程的任务，CoT可能不够有效。此外，CoT可能无法很好地泛化到全新或未预见的任务。

# 8. Inference-time scaling（推理时间扩展）

```
# 参考论文
1. 【Google】Large Language Monkeys: Scaling Inference Compute with Repeated Sampling
2. 【Google】Scaling LLM Test-Time Compute Optimally can be More Effective than Scaling Model Parameters
```

## 8.1 问题背景

- 随着语言模型计算能力的不断增强，其表现有了显著提升。这一进步主要得益于更大规模的模型、更丰富的预训练数据，以及更广泛的后期训练（如通过引入人类反馈进行优化）。然而，在推理阶段，模型通常只进行单次推理，这在一定程度上限制了其性能表现。
- 人类在面对复杂问题时，往往会投入更多时间和精力进行反复思考和调整决策。一个关键问题是：我们能否赋予当前的大语言模型（LLM）类似的能力，即在面对复杂或棘手的输入时，通过有效增加推理计算的次数来提升模型的回答准确性?


> 图片来源 https://x.com/DrJimFan/status/1834279865933332752

## 8.2 解决方案

在推理阶段的计算优化中，提出了多个解决方案来提升生成模型的推理效率和输出质量。我们通过以下两方面的策略来统一思考推理阶段的计算：**提议者（Proposer）与验证者（Verifier）**。

### 8.2.1 总体思路

在推理阶段，通过调整模型在给定提示下生成的预测分布，可以有效利用额外的计算资源，目标是生成比直接从大语言模型（LLM）中采样更优的输出结果。这一过程可以通过两种核心策略实现：一是优化生成的候选答案，二是通过验证器筛选并提升候选答案的质量。

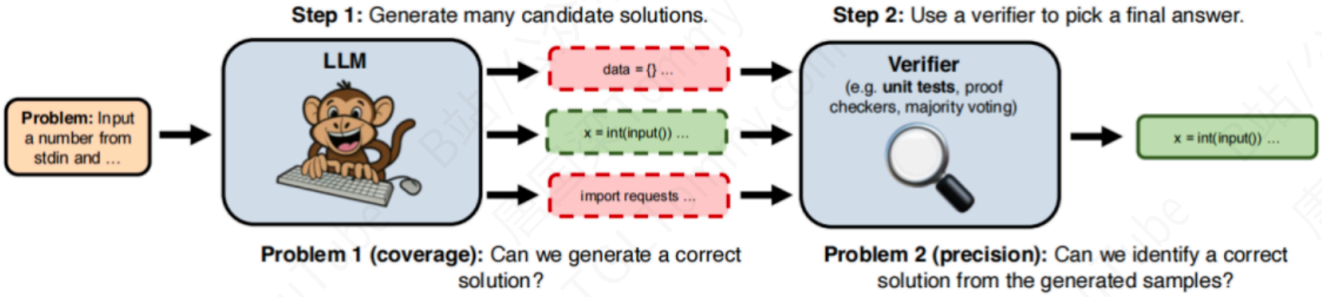



图 1：我们在本文中遵循的重复采样程序。1）我们通过从温度为正的 LLM 中采样，生成许多候选解决方案。2）我们使用领域特定的验证器（例如，代码的单元测试）从生成的样本中选择最终答案。

## 8.2.2 两个关键点

- **输入层面**：通过增加一组额外的标记（tokens），自适应调整模型在提示下的条件分布，进而引导模型更好地生成目标输出。
- **输出层面**：在输出阶段，通过对模型生成的多个候选结果进行采样，并通过验证器筛选最佳答案。验证器通过评估候选解的质量，帮助模型进一步优化输出。

## 8.2.3 增加样本数量与解决率提升

通过增加样本数量（重复采样）可以显著提升模型的解决率，尤其在多个任务和模型上表现明显。研究发现，样本数量与问题解决率呈对数线性关系，表明通过增加推理阶段的计算资源，可以有效提升模型的性能。

特别地，对于简单和中等难度问题，使用较小的模型并增加推理计算时间往往比训练更大规模的模型效果更好。通过额外的推理计算，如验证阶段优化候选答案，模型可以生成更高质量的输出。

## 8.2.4 关键指标

为了评估这些推理优化方法的效果，两个关键指标是：

- **覆盖率**：通过增加样本数量，能成功解决的问题比例。通常使用**pass@k**指标，定义为在生成的k个候选解中至少一个能够解决问题的比例。
- **精度**：从生成的样本集合中，能否识别出正确答案。通过验证器进一步提升样本精度，确保最终输出的正确性。

## 8.2.5 实验结果

- 在编码和证明等领域，增加样本数可以直接提高性能。通过在SWE-bench Lite上应用重复采样，使用DeepSeek-V2-Coder-Instruct模型的解决率从15.9%（1个样本）提升到了56%（250个样本），超过了使用更强大模型单次尝试的43%的最先进水平。
- 在CodeContests等任务中，增加样本数可以将覆盖率从单次尝试的0.02%提升到10,000次尝试的7.1%。

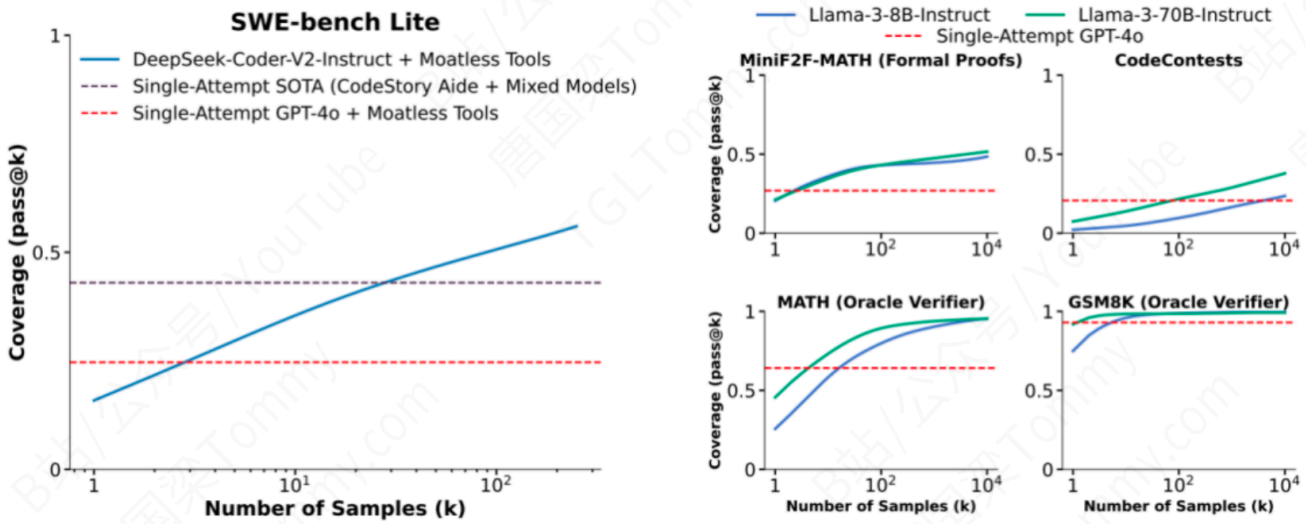



图 2：在五个任务中，我们发现覆盖率（即至少一个生成的样本解决问题的比例）随着样本数量的增加而增加。值得注意的是，使用重复采样，我们能够将一个开源方法在 SWE-bench Lite 上的解决率从 15.9%提升到 56%。

# 8.3 技术细节

## 8.3.1 扩展重复采样

扩展重复采样（scaling repeated sampling）能够在多个任务和样本预算中有效提升覆盖率，不仅可以放大弱模型的能力，还能超越更强大模型的单次推理结果，且更具性价比。

重复采样不仅在Llama系列模型中有效，也在不同模型规模和家族中有效。

- 研究扩展了模型集，包含以下几种模型：
  - **Llama 3 系列**：Llama-3-8B、Llama-3-8B-Instruct、Llama-3-70B-Instruct。
  - **Gemma 系列**：Gemma-2B、Gemma-7B。
  - **Pythia 系列**：Pythia-70M到Pythia-12B（8个模型）。
- 实验表明，重复采样可以显著提升几乎每个模型的覆盖率，尤其是较小的模型。Gemma-2B的覆盖率提升了300倍，从单次尝试的0.02%增加到10,000次尝试的7.1%。在MATH任务中，Pythia-160M的覆盖率从0.27%提升至57%。

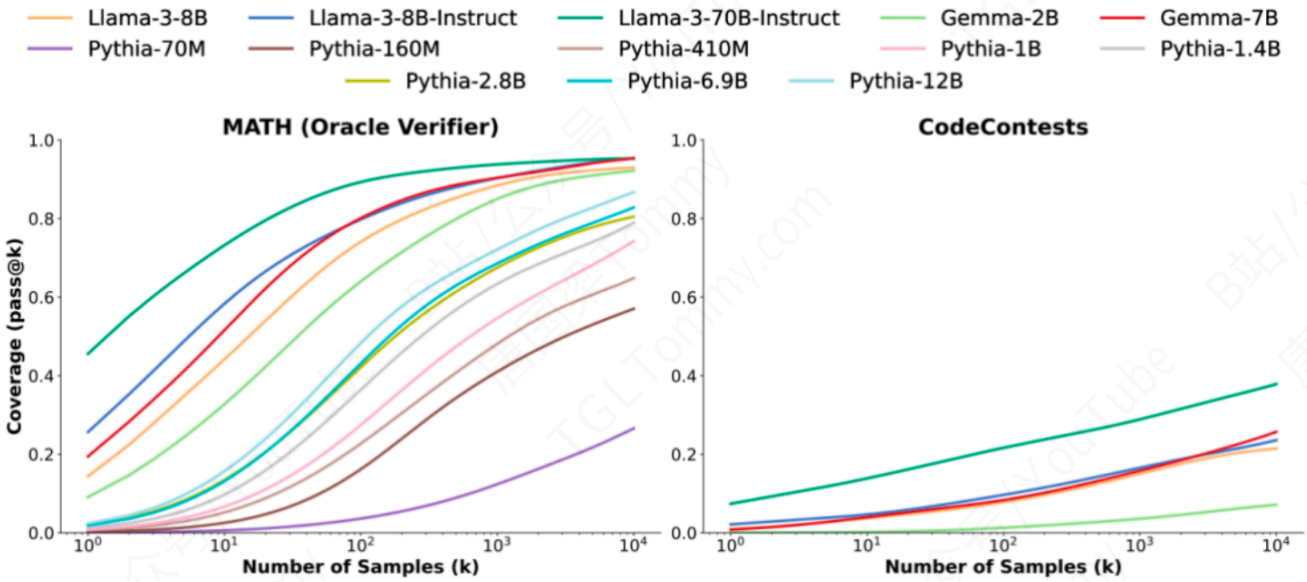



图 3: 通过重复采样扩大推理时间计算，能够在各种模型规模（70M-70B）、模型家族（Llama、Gemma 和 Pythia）以及不同的后训练水平（基础模型和指令模型）上，带来一致的覆盖率提升。

- **成本分析**：重复采样不仅提供了更高的解决率，还比强模型的单次推理更便宜。

| Model | Cost per attempt (USD) | Number of attempts | Issues solved (%) | Total Cost (USD) | Relative Total Cost |
|---|---|---|---|---|---|
| DeepSeek-V2-Coder-Instruct | 0.008 | 5 | 29.62 | 12 | 1x |
| GPT-4o | 0.13 | 1 | 24.00 | 39 | 3.25x |
| Claude 3.5 Sonnet | 0.17 | 1 | 26.70 | 51 | 4.25x |

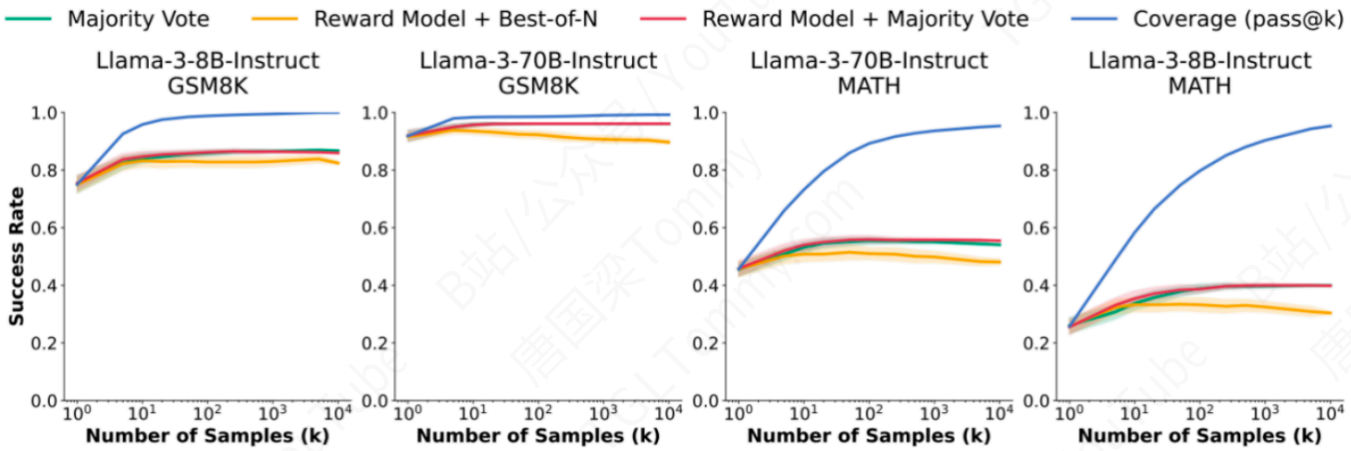
表 1：比较了在 SWE-bench Lite 数据集上使用 Moatless Tools 代理框架的各模型的 API 成本（以美元计）和性能。随着采样数量的增加，开源的 DeepSeekCoder-V2 模型可以以不到三分之一的价格，达到与闭源前沿模型相同的问题解决率。

## 8.3.2 利用重复采样提高精度

**常见的验证方法**：

- **多数投票法**：选取出现频率最高的答案。
- **奖励模型 + 最佳样本法**：使用奖励模型为每个样本打分，选出得分最高的答案。
- **奖励模型 + 多数投票法**：在使用奖励模型对每个样本打分的基础上，再进行多数投票。

使用 **ArmoRM-Llama3-8B-v0.1** 作为奖励模型，并测试了在不同样本数量下的三种方法的效果，发现随着样本数量的增加，成功率逐渐达到一个稳定水平（约100个样本时达到饱和），覆盖率继续增加，超过95%。




- **结论**：重复采样可以显著提高覆盖率，使得在推理时可以放大模型的能力。更弱的模型通过更多的样本可以比更强的模型在更少的尝试中表现更好且成本更低。