GLM-4.5：自主、推理和编码（ARC）基础模型

GLM-4.5团队

智谱AI & 清华大学

(完整作者列表请参阅贡献部分)

摘要

我们介绍了GLM-4.5，一个开源的专家混合（MoE）大语言模型，拥有355B总参数量和32B激活参数量，具备混合推理方法，支持思考和直接响应模式。通过在23T token上进行多阶段训练，以及专家模型迭代和强化学习的全面后训练，GLM-4.5在自主、推理和编码（ARC）任务中表现出色，在TAU-Bench上得分70.1%，在AIME 24上得分91.0%，在SWE-bench验证上得分64.2%。与多个竞争对手相比，GLM-4.5的参数量要少得多，在所有评估模型中排名第三，在自主基准测试中排名第二。我们发布了GLM-4.5（355B参数量）和紧凑版GLM-4.5-Air（106B参数量），以推动推理和自主AI系统领域的研究。代码、模型和更多信息可在https://github.com/zai-org/GLM-4.5获取。

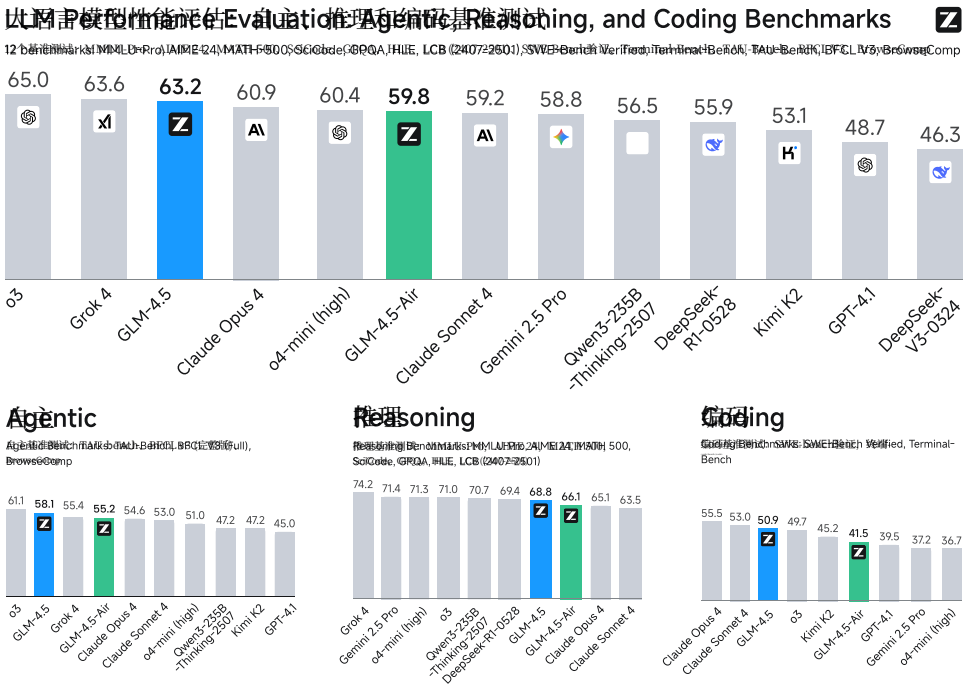


图1：自主、推理和编码（ARC）基准测试的平均性能。总体而言，GLM-4.5排名第三，GLM-4.5-Air紧随其后排名第六。所列模型截至2025年7月28日已评估。

1 简介

大语言模型（LLMs）正迅速从通用知识库 [6; 37; 50; 33; 23] 演变为通用问题解决者。与通用人工智能（AGI）常常关联的最终目标，是创造出在各个领域具备人类水平认知能力的模型。这需要统一掌握复杂问题解决、泛化和自我改进，超越特定任务的卓越表现。

随着大语言模型（LLM）越来越多地融入实际场景，提升实际生产力和解决复杂专业任务的关键在于开发特定的核心能力。我们识别出三种关键且相互关联的能力，作为衡量真正通用模型的指标：**自主**能力用于与外部工具和现实世界交互；在数学和科学等领域解决多步问题的复杂**推理**能力；以及用于应对现实世界软件工程任务的先进**编码**技能。尽管OpenAI的o1/o3 [18]和Anthropic的Claude Sonnet 4等最先进的专有模型在特定的ARC领域（例如数学推理或代码修复 [20]）已展现出突破性性能，但一个能在所有三个领域都表现出色的单一、强大的开源模型仍然难以寻觅。

本文介绍了两款新模型：GLM-4.5和GLM-4.5-Air，旨在统一所有不同能力。新模型在各个方面都优于现有的开源LLM模型 [13; 34; 47]，在自主、推理和编码任务中均有显著提升。GLM-4.5和GLM-4.5-Air都具备混合推理模式：思考模式用于复杂推理和自主任务，非思考模式用于即时响应。GLM-4.5是我们首款MoE模型，总参数量为355B，激活参数量为32B。GLM-4.5在以下ARC基准测试中展现出强大性能：

- **自主：** GLM-4.5 在 TAU-Bench 上得分 70.1%，在 BFCL v3 [26]，上得分 77.8%，与 Claude Sonnet 4 相当。对于网络浏览代理，GLM-4.5 在 BrowseComp [45], 上的得分为 26.4%，明显优于 Claude Opus 4 (18.8%)，接近 o4-mini-high (28.3%)。
- **推理：** GLM-4.5在一系列具有挑战性的推理基准测试中表现出色，在AIME 24上达到91.0%，在GPQA [30], 72上达到79.1%，在LiveCodeBench (2407-2501) [19], 上达到9.9%，在HLE (人类最后考试) [28]上达到14.4%。
- **编码:** GLM-4.5在SWE-bench上得分64.2%，在Terminal-Bench上得分37.5%，优于GPT-4.1和Gemini-2.5-pro，接近Claude Sonnet 4。

GLM-4.5-Air 是一个参数量为 106B 的较小 MoE 模型。它在 100B 规模的模型中实现了显著跨越，与 Qwen3-235B-A22B [47] 和 MiniMax-M1 [7] 相当或超越。

在图 1 中，我们展示了在自主、推理和编码（ARC）任务上 12 个基准测试的平均性能。总体而言，GLM-4.5 位列 **第**三名，GLM-4.5-Air 位列 **第六**名。在自主任务中，GLM-4.5 位列第 2 名，仅次于 OpenAI o3。在编码任务中，GLM-4.5 位列第 3 名，接近 Claude Sonnet 4。请注意，GLM-4.5 具有高度参数效率，其参数量仅为 DeepSeek-R1 [13] 的一半，且为 Kimi K2 [34] 的三分之一。在图 2 中，我们报告了不同开源模型在 SWE-bench 验证上的得分与模型参数的关系，其中 GLM-4.5 和 GLM-4.5-Air 位于帕累托前沿。更多评估结果详见第 4 节。

GLM-4.5 和 GLM-4.5-Air 都可以在 Z.ai、BigModel.cn 上使用，同时它们也是开源模型，托管在 https://huggingface.co/zai-org/GLM-4.5 上。我们还开源了一个评估工具包，发布在 https://github.com/zai-org/glm-simple-evals，以确保我们基准测试结果的复现性。

2 预训练

2.1 架构

在GLM-4.5系列中，我们采用了MoE架构，这提高了训练和推理的计算效率。我们采用无损平衡路由 [40] 和用于MoE层的Sigmoid门 [23]。与DeepSeek-V3 [23] 和Kimi K2 [34]，不同，我们减少了模型的宽度（隐藏维度和路由专家的数量），并增加了其高度（层数），因为我们发现更深层的模型表现出更好的推理能力。在自注意力组件中，

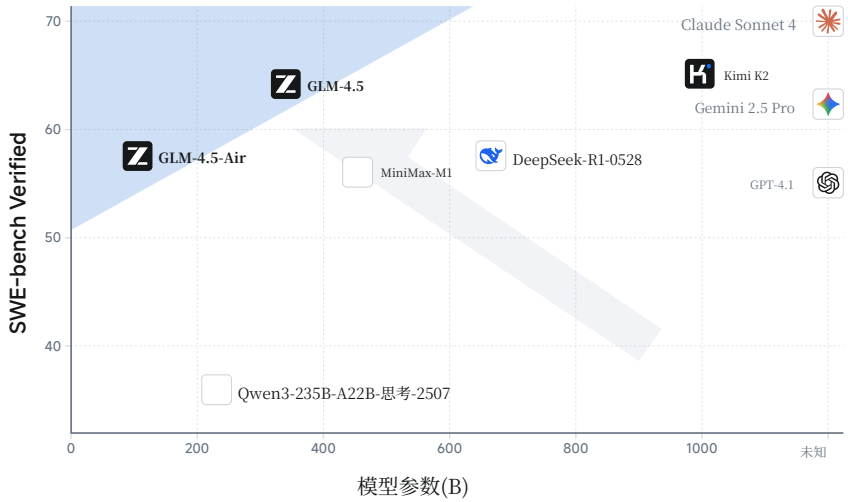


图2：SWE-bench验证分数与模型参数对比。专有模型在右侧列出的未知位置。

我们采用带部分RoPE的分组查询注意力。此外，我们使用了2.5倍更多的注意力头（对于5120隐藏维度的模型为96个头）。反直觉地，虽然这种增加的头数与头数较少的模型相比并未提高训练损失，但它始终提高了在MMLU和BBH等推理基准测试上的性能。我们还集成了QK-Norm [15]来稳定注意力logits的范围。对于GLM-4.5和GLM-4.5-Air，我们添加了一个MoE层作为MTP（多标记预测）层 [12]，以支持推理过程中的推测解码。

表1：GLM-4.5和GLM-4.5-Air的模型架构。在统计参数时，对于GLM-4.5和GLM-4.5-Air，我们包含MTP层的参数，但不包含词嵌入和输出层。

模型	GLM-4.5	GLM-4.5-Air	DeepSeek-V3	Kimi K2
# 总参数量	355B	106B	671B	1043B
# 激活参数量	32B	12B	37B	32B
# 密集层	3	1	3	1
# MoE层	89	45	58	60
# MTP层	1	1	1	0
隐藏维度	5120	4096	7168	7168
密集中间维度	12288	10944	18432	18432
MoE中间维度	1536	1408	2048	2048
注意力头维度	128	128	192	192
# 注意力头	96	96	128	64
# 键值头	8	8	128	64
# 专家（总数）	160	128	256	384
# 专家活跃数（每token）	8	8	8	8
# 共享专家	1	1	1	1
QK-Norm	Yes	No	No	No

2.2 预训练数据

Our 预训练语料库包含来自网页、社交媒体、书籍、论文和共同体的文档de 仓库存储库。我们精心设计了针对不同来源的数据处理管道。

Web 我们的大部分预训练文档是来自互联网的英语和中文网页。受 Nemotron-CC [32]，启发，我们将爬取的网页按不同的质量分数分成不同的桶。我们从质量分数较高的桶中进行文档上采样

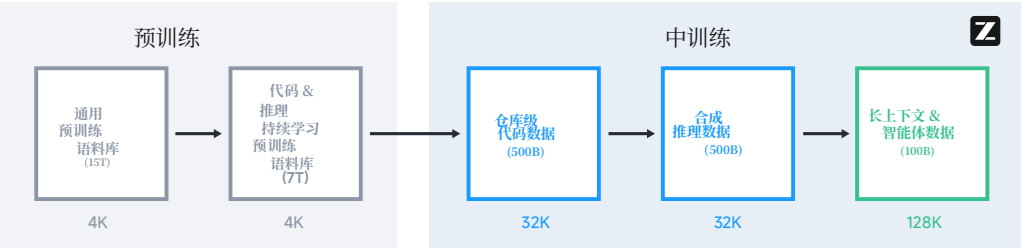


图3：GLM-4.5的预训练和中训练阶段。我们采用多阶段训练配方，并将序列长度从4K扩展到128K。

并丢弃质量分数最低的桶中的文档。质量分数最高的桶在预训练期间贡献了超过3.2个epoch。通过这种方式，预训练语料库可以强调推理任务的高频知识，并提高长尾世界知识的覆盖率。我们还发现大量从模板自动生成的相似网页，并给予了高分。这些网页无法通过MinHash去重删除。我们额外应用了SemDedup [1] 管道，基于文档嵌入来移除这些相似网页。

多语言为了支持更多自然语言，我们在预训练语料库中包含了多语言文档。多语言语料库来自我们爬取的网页和Fineweb-2 [27]。我们应用了一个质量分类器来评判文档的教育效用，并对高质量的多语言文档进行上采样。

代码我们从GitHub和各种代码托管平台精选了源代码数据。代码语料库经过初步基于规则的过滤，然后使用特定语言的质量模型进行分类，将样本分为三个等级：高质量、中等质量和低质量。在训练过程中，我们对高质量代码进行上采样，同时排除低质量样本。此外，所有源代码数据都应用了Fill-In-the-Middle [5] 训练目标。对于代码相关的网页文档，我们从文本预训练语料库中采用两阶段检索过程。文档最初根据两个标准进行选择：存在HTML代码标签，或由一个经过训练以检测代码相关内容的FastText [22] 分类器识别。随后，检索到的文档使用专用模型进行质量评估，并根据与源代码相同的质量采样策略将其分类为高、中或低质量。最后，采用细粒度解析器对选定的网页重新解析，以更好地保留代码的格式和内容。

数学 & 科学 为提升推理能力，我们从网页、书籍和论文中收集与数学和科学相关的文档。我们应用大型语言模型根据数学和科学教育内容的比例对候选文档进行评分，并训练一个小型分类器来预测分数。预训练语料库中评分高于某一阈值的文档会被进行上采样。

GLM-4.5的预训练过程分为两个阶段。在第一阶段，模型主要在来自网页的通用文档上进行训练。在第二阶段，我们从GitHub和与编码、数学、科学相关的网页中上采样源代码。

2.3 中训练：提升推理 & 自主能力

预训练后，我们添加了几个阶段来进一步提升模型在重要应用领域的性能。与传统在大型通用文档上进行预训练不同，这些训练阶段利用中等规模的特定领域数据集，包括指令数据。因此，我们将这些训练阶段称为中训练，具体包括以下内容。

仓库级代码训练在本次训练阶段，我们将来自同一仓库的连接代码文件添加进来以学习跨文件依赖。为了提升模型的软件工程能力，

我们还包含 GitHub 上的模型筛选的 issue、pull request（PR）和 commit，并将相关的 issue、PR 和 commit 连接成一个上下文，同时将 commit 以类似 diff 的格式组织。我们将训练序列长度从 4K 扩展到 32K 以包含大型仓库。

合成推理数据训练在这个阶段，我们为数学、科学和编程竞赛添加合成推理内容。我们从网页和书籍中收集大量与推理任务相关的问答题和答案，并使用推理模型合成推理过程。

长上下文与代理训练为了进一步提升模型的长期上下文性能，我们将训练序列长度从32K扩展到128K，并从预训练语料库中上采样长文档。在这个阶段，我们还加入了大规模的合成代理轨迹。

在图3中，我们展示了预训练和中训练的完整阶段。预训练中最大序列长度保持在4,096，而中训练时则从32,768扩展到131,072。在预训练期间，我们没有使用最佳拟合打包 [11]，因为随机截断是一种很好的预训练文档数据增强策略。对于中训练的数据集，我们应用了最佳拟合打包，以避免截断推理过程或仓库级代码。

2.4 超参数

我们采用了Muon优化器 [21; 24] 除词嵌入、偏差和RMSNorm的权重之外，所有参数都使用该优化器。对于超参数，我们将牛顿-舒尔茨迭代步数 N 设置为5，动量 μ 设置为0.95，并将Muon的更新RMS缩放至0.2。我们观察到Muon优化器可以加速收敛并容忍更大的批处理大小。我们使用了余弦衰减学习率调度，而不是预热-稳定-衰减（WSD）调度 [17]。我们的早期实验表明，使用WSD调度训练的模型在通用基准测试（SimpleQA、MMLU）上的表现更差，这表明在稳定阶段存在欠拟合。学习率经历了从0到2.5e-4的预热阶段，以及到2.5e-5的衰减阶段，直至中训练结束。我们使用了批处理大小预热策略，在训练前500B个token时，批处理大小从16M token逐渐增加到64M token，并在剩余训练中保持不变。对于正则化，我们将权重衰减率设置为0.1，并且没有使用dropout。我们在预训练期间将最大序列长度设置为4,096，并在中训练阶段根据图3将其扩展到32,768和131,072。当将序列长度扩展到32K时，我们还调整了RoPE的基础频率从10,000到1,000,000，以获得更好的长上下文建模能力。对于无损平衡路由，我们将偏差更新率设置为前15T token为0.001，其余token为0.0。我们还应用了权重为0.0001的辅助序列级平衡损失，以避免任何单个序列内的极端不平衡。MTP损失权重 λ 在前15T token时设置为0.3，其余token设置为0.1。

3 专家模型迭代：后训练

我们将后训练过程分为两个不同阶段。在阶段1（专家训练）中，我们构建专注于三个领域的专家模型：推理、代理和通用聊天。在阶段2（统一训练）中，我们采用自蒸馏技术整合多个专家，最终交付一个能够通过推理和直接响应模式生成回复的综合性模型。

3.1 监督微调

我们在 Stage 1 (专家训练) 和 Stage 2 (统一训练) 的开始都进行监督微调（SFT）。在专家训练阶段，SFT 的主要作用是提供冷启动，赋予模型基本的聊天、推理和工具使用能力，这些能力随后可以在后续的专家强化学习训练中进一步强化，以实现性能提升。在统一训练阶段，SFT 的目的是将不同专家模型的能力提炼到一个混合推理通用模型中，使其能够处理不同类型的任务。

冷启动 SFT 在冷启动阶段，我们使用包含扩展思维链（CoT）响应的小规模监督微调（SFT）数据集。这种方法确保每个专家模型在强化学习阶段之前都具备足够的基础能力。

整体微调 在整体微调阶段，我们从先前训练的专家模型中收集数百万个样本，涵盖推理任务（数学、代码、科学等）、通用聊天（写作、翻译、摘要、闲聊等）、自主任务（基本工具使用、编码能力尤其是真实项目开发等）以及长上下文理解任务，并使用最大上下文长度为128 K token的基础模型进行训练。通过从不同专家的输出中提取知识，模型学习为每个任务应用最有效的长CoT推理来获得准确答案。特别是，认识到对于需要快速响应的某些领域（如闲聊）来说，冗长的思考过程是不必要的，我们精心平衡了包含完整推理的训练数据和缺乏明确思考过程的训练数据。这种方法使模型能够在反思和即时响应模式之间切换，从而创建一个混合推理模型。此外，我们发现以下策略在准备微调数据以获得最佳性能方面很有帮助。

减少函数调用模板中的字符转义 尽管在当代实现中，函数调用参数主要以JSON格式表示，但当这些参数包含代码片段时，一个重大挑战会出现。在这种情况下，代码中相当一部分字符需要转义，迫使模型生成大量的转义字符，从而增加了模型的学习负担。虽然这个问题对主要设计用于通用聊天的模型影响不大，但对于以函数调用为核心能力的自主基础模型来说却是一个非平凡的挑战。为了缓解这一限制，我们提出了一种新的函数调用模板，该模板将函数调用键和值封装在类似XML的特殊标记中。这种方法大大减少了代码片段中字符转义的需求，因为绝大多数代码可以以其原生形式表示而无需转义。实验结果表明，所提出的函数调用模板在减少转义的同时，并未影响函数调用执行的性能。以下示例（图4）说明了我们提出的函数调用模板的结构。详细的代码实现可以在我们的开源仓库中找到。

```
<|系统|># 工具

您可以调用一个或多个函数来协助处理用户查询。

您在 <tools></tools>XML标签中提供了函数签名：<tools>

{"name": "获取_天气"}         "description": "获取特定日期的城市天气。",                                    "parameters": {"
type": "object",       "properties": {"城市": {"type": "string",}}             "description": "获取天气的城市,"
在中文。"},          "日期": {"type": "string",}          "description": "YYYY-MM-DD格式日期。"}}                    "必需":
["城市"]}}
</工具>

对于每个函数调用，以以下XML格式输出函数名称和参数：
<工具_调用>{函数名称}
<参数_键>{参数键-1}</参数_键>
<参数_值>{参数值-1}</参数_值>
<参数_键>{参数-键-2}</参数_键>
<参数_值>{参数-值-2}</参数_值>
...
</工具_调用><|系统|>
你是一个有帮助的助手.<|用户|>
今天是2024年6月26日。你能帮忙查看明天北京和上海的天气吗<|
助手|>
<想>用户想查询北京和上海明天的天气。我需要调用
get_weather函数分别查询北京和上海.</想>
我将调用get_weather函数查询北京和上海的天气。
<工具_调用>get_weather
<参数_key>城市</参数_key>
<参数_value>北京</参数_value>
<arg_key>日期</arg_key>
<arg_value>2024-06-27</arg_value>
</工具_调用>
<工具_调用>获取_天气
<arg_key>城市</arg_key>
<arg_value>上海</arg_value>
<arg_key>日期</arg_key>
```

<arg_value>2024-06-27</arg_value></tool_调用><|观察|><tool_响应>{"城市": "北京", "日期": "2024-06-27", "天气": "晴朗", "温度": "26C"}</tool_响应><tool_响应>{"城市": "上海", "日期": "2024-06-27", "天气": "阴天", "温度": "29C"}</tool_响应><|助手|><思考>我已经分别获取了北京和上海的天气查询结果，可以直接回复用户。</思考> 北京明天将是晴朗的，气温为26摄氏度。上海的天气是阴天，气温为29摄氏度。<|用户|>

图4：函数调用模板的一个示例。

拒绝采样 从专家模型中采样时，我们采用一个包含多阶段的综合过滤管道，包括：(1) 移除重复、过短或截断的样本，以及那些不符合有效推理格式的样本；(2) 对具有客观答案的样本进行正确性验证；(3) 利用奖励模型过滤主观问题的响应；(4) 对于工具调用场景，确保遵循正确的工具调用协议，并验证轨迹达到预期的终止状态。

提示选择和响应级缩放 过滤具有挑战性的提示并对它们进行响应缩放被证明是有效的。我们基于响应长度移除了排名后50%的提示，尽管仅使用一半数据训练，数学和科学任务仍提升了2%-4%。值得注意的是，我们发现将这些难提示应用响应缩放可以带来进一步收益。为每个提示生成四个响应额外提升了1%-2%。

自动自主SFT数据构建自主SFT数据的构建涉及四个步骤：1. 自主框架和工具收集：我们收集一套自主框架和真实世界的工具API及MCP服务器，同时利用大语言模型自动构建和模拟一批工具。2. 任务合成：基于这些框架和工具，我们自动合成一系列自主任务。一方面，对于相对成熟的框架，我们利用大语言模型理解其功能并自动生成相关查询或任务。另一方面，对于更零散或异构的工具，我们首先选择一个代表性子集，并同样采用大语言模型构建关于该子集的任务。这些任务涵盖单步和多步工具调用场景。3. 轨迹生成：对于每个合成的任务，我们利用现有的大语言模型生成工具调用轨迹。此外，通过将大语言模型作为用户模拟器，多步工具调用任务被转换为涉及多轮对话的轨迹。4. 质量过滤：对于每个轨迹，多个裁判智能体被用于评估任务是否完成。只有成功的轨迹被保留。

3.2 推理强化学习

推理强化学习专注于提升模型在需要逻辑推理、结构化问题解决和可验证准确性的领域中的能力。这包括数学、代码生成和科学推理等关键领域。这些任务的一个显著特征是其奖励信号的高精度，因为正确性通常可以通过编程或客观清晰的方式确定。在这些领域的掌握不仅对于提升模型的原始智能至关重要，而且也是更复杂、多步骤的自主行为的基础构建模块。为了应对推理强化学习中的独特挑战和机遇，我们开发了一套专门的技术来有效训练我们的模型。以下详细介绍了这些方法，旨在解决训练效率、样本多样性和数据质量等问题。我们的整体强化学习算法基于GRPO [31] 框架，排除了KL损失项。本节显示的比较曲线基于我们的较小实验模型，而非GLM-4.5。

基于难度的课程学习 在强化学习中，模型的熟练度会随时间演变，从而与静态训练数据产生不匹配。在后期阶段，随着模型能力的提升，过于简单的数据可能导致所有奖励都为1的 rollout。相反，在早期阶段，过于困难的数据往往会导致所有奖励都为0的批次。在两种情况中

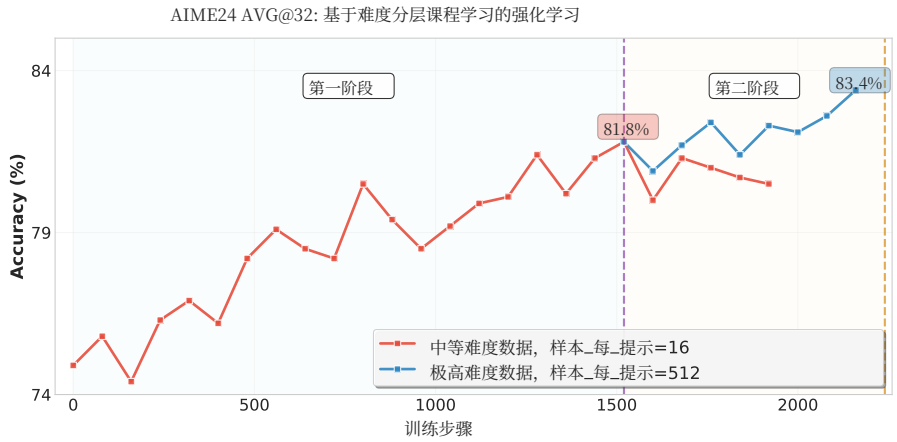


图 5: **基于难度的两阶段课程在 AIME’24 上的有效性。** 蓝线(我们的方法) 在第二阶段切换到极难问题 (pass@8=0, pass@512>0)，显示出持续改进。 红线 (基线) 继续使用中等难度问题并进入平台期。

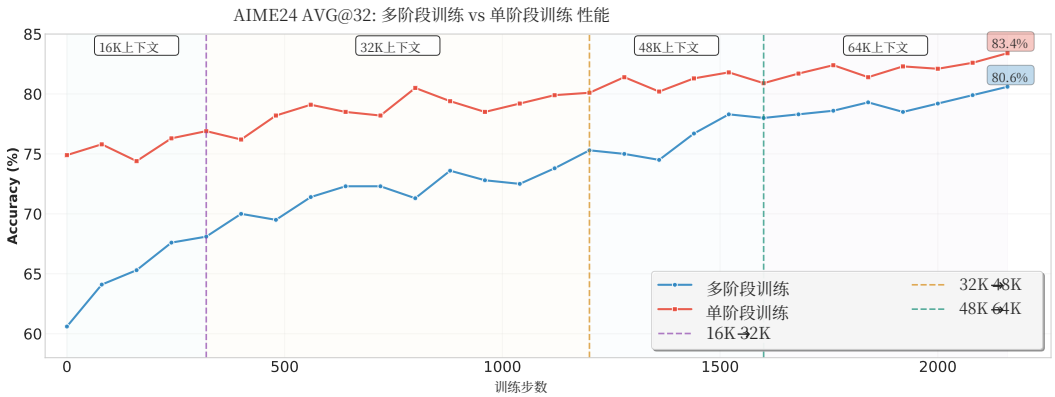


图 6: **64K 上下文长度下的单阶段与多阶段强化学习。** 红线(64K 单阶段) 实现了优异的性能。而蓝线 (逐步增加长度的多阶段) 在早期阶段出现不可逆的性能下降，限制了其最终性能。

场景中，奖励方差缺乏提供了无用的梯度信号，严重阻碍了训练效率。为应对这一挑战，我们采用基于难度的两阶段课程对强化学习进行训练。通过在较小模型上进行受控实验验证了本方法及其他下文讨论策略的有效性，这允许快速迭代和精确的消融研究。如图5所示，这种两阶段方法使模型能够持续超越其性能上限。关键的是，为保持高信号质量并减少噪声，第二阶段使用的所有问题都严格来源于已验证正确答案的池。

64K输出长度单阶段RL先前研究 [25] 已表明，应使用逐步增加最大输出长度的多阶段方法进行RL。然而，我们的实验表明，这种多阶段方法不如直接在64K最大目标长度进行单阶段RL有效。由于初始监督微调（SFT）已使模型适应生成64K长度响应，引入最大长度更短RL阶段会导致模型“遗忘”其长上下文能力。这通常会导致性能出现显著且不可逆的下降，因为模型的平均输出长度会减小。这种退化在最终64K长度RL阶段难以恢复，从而限制了进一步改进。我们的实验证实了这一观察：如图6所示，直接在完整64K长度应用RL持续推动模型极限并产生更好的性能。

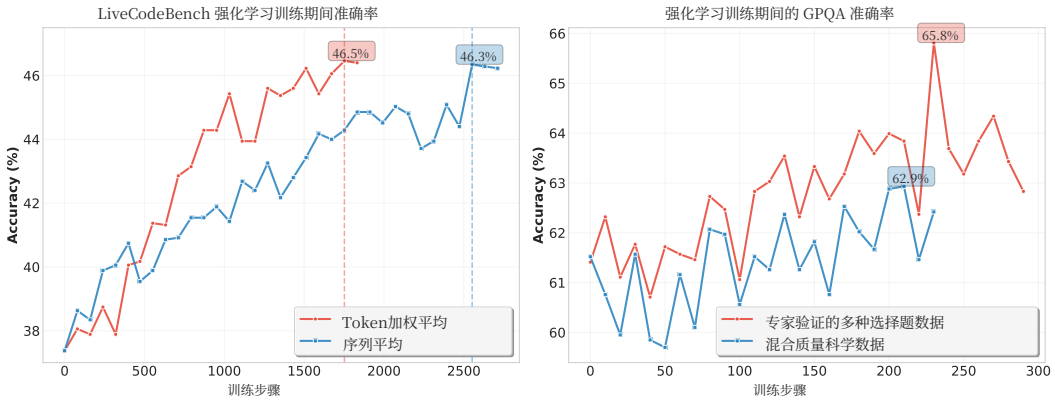


图 7: **代码与科学强化学习的消融研究。(左)** 代码强化学习的损失计算方法对比。token加权平均损失 方法相比序列平均损失 基线收敛更快，加速了训练过程。 序列平均损失 基线，加速了训练过程。 **(右)** 在 GPQA-Diamond 基准上对科学强化学习的数据源进行消融。仅使用一小部分高质量、专家验证的简答题进行训练，性能最佳，显著优于使用混合质量数据的训练。

动态采样温度 在强化学习中，采样温度是控制轨迹多样性的一个关键参数。温度太低会导致收敛、探索性不足的输出，而温度太高则会引入低质量、噪声较大的样本，从而降低模型精度和训练效率。使用固定的采样温度是不理想的，因为它无法适应策略分布变得更加集中（即熵更低）的情况，通常导致后期探索不足。因此，我们提出动态调整采样温度，以在精度和探索之间保持健康的平衡。具体来说，当rollouts的平均奖励稳定时，我们将其识别为收敛阶段，并增加采样温度以鼓励更大的多样性。为了降低引入过多噪声的风险，我们实施了一种质量控制机制：我们在一系列温度下，定期在保留的验证集上评估模型性能。然后，将下一阶段的训练温度设置为不会导致性能下降超过1%的当前最优值 [2]。

代码与科学强化学习 与数学相比，代码和科学领域的强化学习在文献中受到的关注较少。我们在这些领域进行了广泛的受控强化学习实验，并得出以下经验性结论。对于 **代码强化学习**，我们发现损失计算的选择对训练效率至关重要。如图 7 (左) 所示，采用 token 加权平均损失比传统的序列平均损失更有利。token 加权方法提供了更细粒度和更稳定的梯度信号，从而显著加快收敛速度。该方法还有助于缓解序列级奖励中固有的长度偏差，并有效抑制训练过程中生成过于简单或重复的“基本情况”样本。对于 **科学强化学习**，我们在 GPQA-Diamond 基准上的发现表明，数据质量和类型是至关重要的因素。如图 7 (右) 所示，仅使用专家验证的简答题进行强化学习，性能显著优于使用混合质量或未验证数据的训练。这一结果强调了即使对于像简答题这样格式简单的任务，严格过滤强化学习数据池，仅包含高质量、具有挑战性的实例，对于有效模型改进至关重要。

3.3 自主强化学习

人类反馈强化学习（RLHF）帮助语言模型更忠实地遵循人类指令。将RL应用于数学和编程竞赛进一步揭示了其在结果可客观验证的任务上具有强大的推理能力和有利的缩放行为。基于这些见解，我们专注于自主设置——特别是网络搜索和代码生成代理——其中每个行动或答案都可以自动检查。这种内置的可验证性提供了密集、可靠的奖励，使我们能够更有效地扩展RL训练。

3.3.1 数据收集与合成（针对代理）

对于网络搜索任务和开放域信息检索，我们开发了一个数据合成管道，该管道生成需要跨多个网络来源进行多步推理的具有挑战性的问答对。该语料库旨在提高GLM在互联网上发现难以捉摸、交织事实的能力。数据集构建融合了两种方法：(1)一个由知识图谱多跳推理驱动的自动化管道，以及(2)从多个网页中通过人类参与循环提取和选择性模糊内容，以准备强化学习训练信号。

对于软件工程任务，我们整理了大量 GitHub 拉取请求和问题，以创建一个包含用户提示和可执行单元测试的、真实的软件开发基准。所有评估都在一个加固的沙盒内进行，该沙盒包含分布式系统，提供水平可扩展性和强隔离保证。

3.3.2 通过强化学习和迭代自蒸馏突破极限

我们采用组内策略优化算法进行强化学习训练。对于每个问题 x，我们从先前的策略 π_{old} 中采样 K 个代理轨迹 $\{y_1, \ldots, y_k\}$，并针对以下目标优化模型 π_θ：

$$L_{\text{RL}}(\theta) = \mathbb{E}_{x\sim\mathcal{D}}\left[\frac{1}{K}\sum_{i=1}^{K}\left(r(x, y_i) - \bar{r}(x)\right)\right],$$

其中 $\bar{r}(x) = \frac{1}{k}\sum_{i=1}^{k} r\big(x,\ y_i\big)$ 是采样响应的平均奖励。需要注意的是，仅使用模型生成的标记进行优化，环境反馈在损失计算中被忽略。

过程行动格式惩罚的成果监督对于网络搜索任务，我们使用最终答案的准确度作为整个智能体轨迹的奖励。对于编码智能体，我们主要使用具有可验证测试用例的 SWE 数据进行强化学习训练。我们的实验表明，在网络搜索和 SWE 任务上进行强化学习训练，可以提升智能体在其他任务和基准测试（例如通用工具使用和编码任务如 Terminal-Bench）上的泛化性能。此外，我们应用过程行动格式惩罚，以确保模型生成正确的工具调用格式。如果模型在生成智能体轨迹时未能正确生成工具格式，过程将被中止，轨迹将获得零奖励。

迭代蒸馏 由于在代理任务上进行强化学习训练耗时较长，我们采用自蒸馏方法，在重新开始在此改进模型上进行强化学习训练之前，迭代提升SFT冷启动模型的性能。具体来说，我们首先对初始冷启动模型进行强化学习训练以提升代理性能。一旦训练达到一定的步数或进入平台期，我们通过用强化学习训练模型生成的响应替换原始冷启动数据来进行自蒸馏，从而创建一个更优的SFT模型。然后我们对这个增强模型进行进一步的强化学习训练，逐步提高训练难度。这种迭代策略使我们能够高效地提升强化学习训练模型的性能上限。

通过交互轮次扩展测试时计算 对于代理任务，我们观察到随着与环境的交互轮次增加，性能显著提升。与推理模型中的测试时扩展（该扩展扩展输出令牌）相比，代理任务通过持续与环境交互来利用测试时计算，例如搜索难以找到的网页信息或为自我验证和自我纠正编码任务编写测试用例。图8显示，随着浏览努力的改变，准确率与测试时计算平滑扩展。

3.4 通用强化学习

通用强化学习旨在全面提升模型的整体性能，解决潜在问题并增强关键能力。我们的方法论核心是一个多源反馈系统，整合了基于规则的反馈、人类反馈（RLHF）和基于模型的反馈（RLAIF）。这种混合框架提供了更稳健的训练信号，并使我们能够利用每个来源的独特优势：自动化规则的精确性、人类标注员的细致判断以及AI驱动评估的可扩展性。

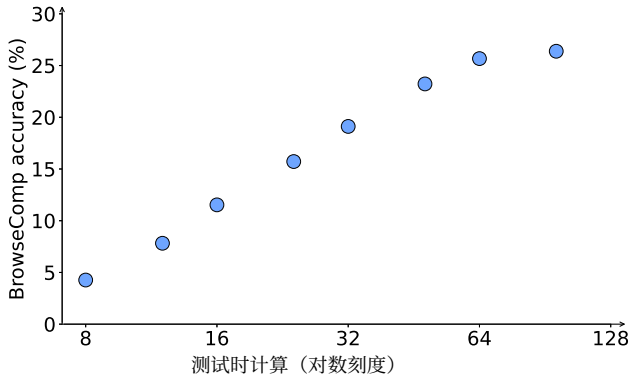


图8：BrowseComp的交互回合扩展

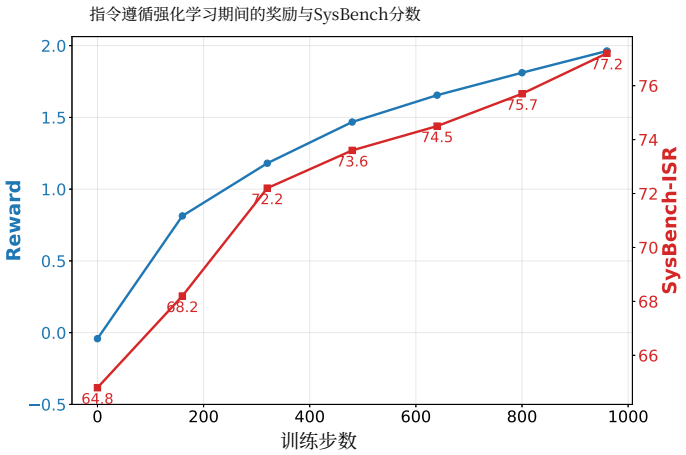


图9： **不含其他通用强化学习任务的指令遵循强化学习训练曲线**。 在GRPO训练期间，指令遵循性能（SysBench-ISR）随着奖励的增加而提升。在约1,000个训练步数之前，我们没有观察到明显的奖励攻击证据。

整体强化学习 整体强化学习旨在跨不同领域实现广泛的性能提升。为此，我们首先构建了一个包含约5,000个提示的平衡数据集，涵盖7个主要类别、33个次要类别和139个三级类别。整体强化学习的奖励信号源自人类和AI的反馈。对于人类反馈，我们在偏好标注上训练奖励模型。标注员通过综合评估多个维度（如指令遵循、安全性和事实准确性）来比较模型响应并分配偏好标签。对于模型反馈，我们设计了不同的评分标准，这些标准取决于提示是否具有客观的ground-truth答案。合并这两个反馈来源产生了更可靠和表达力更强的奖励信号，减轻了每种方法的固有局限性。

指令遵循强化学习 指令遵循强化学习提升了模型理解和满足复杂指令的能力。为此，我们创建了一个细粒度的分类体系，包含7种主要约束和151种次要约束，涵盖内容要求、格式规则等。基于此分类体系，我们汇编了一个包含挑战性指令的专用训练集，以覆盖所有约束类型。反馈系统由确定性验证规则、训练好的奖励模型和评论模型组成。这种混合反馈系统的鲁棒性在GRPO训练期间被证明至关重要。我们观察到奖励攻击得到缓解，使得策略模型能够在指令遵循方面实现持续和稳定的改进，如图9所示。

函数调用强化学习 函数调用强化学习分为逐步基于规则的强化学习和端到端多轮强化学习。我们将逐步基于规则的强化学习直接整合到我们的通用强化学习框架中，由于

到他们的相似输出长度和收敛速度。对于端到端的multi-turn RL，我们首先训练专门的专家模型，然后将这些专家蒸馏到主模型中。

• **逐步基于规则的强化学习**：对于具有明确工具调用流程的任务，我们在训练数据中为每个步骤/回合标注真实的函数调用。给定任务以及先前步骤/回合的函数调用，模型被训练生成下一个助手响应，该响应可以是函数调用或对用户的回复。使用基于规则的奖励，我们引导模型在连续回合中做出正确的函数调用。相应地，我们设计了以下严格的奖励函数：

$$\text{Reward} = \begin{cases} 1, & \text{if } \texttt{FormatCorrect}(a_t) \text{ and } \texttt{Match}(a_t, a_t^*) \\ 0, & \text{otherwise} \end{cases}$$

这里， a_t 表示模型生成的第t个函数调用， a_t^* 是相应的真实函数调用。只有当 a_t 格式正确且与真实值完全匹配（包括名称、参数和每个字段）时，才会给予1分的奖励。否则，奖励为0。这种严格的奖励规则不仅引导模型生成正确的函数调用，还强烈规范输出格式，提高模型在实际交互中的可用性和鲁棒性。

• **端到端多回合强化学习**：逐步基于规则的强化学习将任务分解为静态、预定的决策流程。在这个过程中，模型缺乏与环境的动态交互，无法自主探索、规划或处理复杂情况，从而使其实际问题解决能力有限。为解决这些问题，我们引入端到端多回合函数调用强化学习，其中模型首先生成完整轨迹，然后根据任务完成情况进行奖励。通过这种方式，模型可以通过工具反馈的持续试错来优化其动作策略，显著增强其自主规划和决策能力。具体而言，端到端多回合函数调用强化学习考虑了两种复杂任务：1. 单回合多步任务：模型需要执行多步函数调用并与环境交互以完成任务。我们使用基于MCP服务器自动合成复杂任务，以及一些具有可运行环境的开源自主数据集，例如Agentgym [46]。2. 多回合多步任务：除了与工具执行环境交互外，模型还需要与LLM模拟的用户代理交互以获取完整任务信息并完成整体任务。端到端多回合函数调用强化学习的奖励计算如下：

$$\text{奖励} = \begin{cases} 1, & \text{如果格式正确}(a_1, \ldots, a_T) \text{并且任务完成}(I, o_0, a_1, o_1, \ldots, a_T, o_T) \\ 0, & \text{否则} \end{cases}$$

这里， I 指的是原始复杂任务， a_t 是第t次函数调用， o_t 是工具反馈或用户信息。 $\text{TaskCompleted}(I, o_0, a_1, o_1, \ldots, a_T, o_T)$表示任务是否完成，这由环境根据预定义规则或由大语言模型裁判代理决定。

病理强化学习作为后训练的最终阶段，通用强化学习需要纠正潜在问题，如语言混杂、过度重复和格式错误。尽管在上述通用强化学习任务中惩罚这些行为是有效的，但这些病理现象（通常占输出不到1%）的发生率较低，使得这种优化策略样本效率低下。因此，我们通过识别高度可能触发这些病理行为的提示，构建了一个针对病理强化学习的特定数据集。在这个数据集上进行训练，使我们能够施加高效的惩罚，进一步降低这些问题行为的残余错误率。

3.5 强化学习基础设施

我们的强化学习基础设施基于Slime[1]，这一我们开发的开源框架构建。该框架经过精心设计，包含多项关键优化，以提升灵活性、效率和可扩展性。

灵活混合训练与数据生成架构 我们基础设施的核心特性在于其支持高度灵活的训练范式和数据生成策略，且这一切都在单一统一系统中完成。这种设计使我们能够通过

[1] https://github.com/THUDM/slime

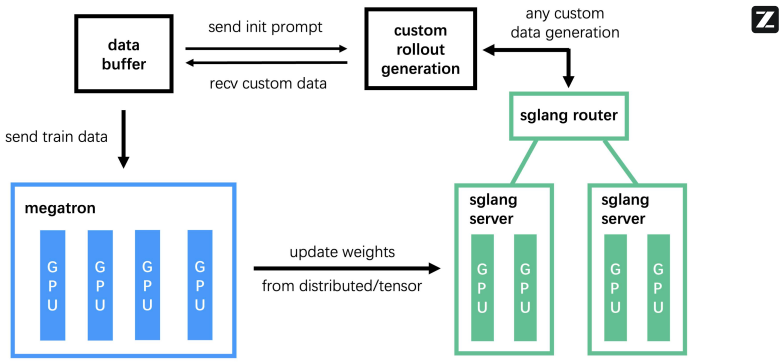


图 10：Slime 强化学习基础设施概述。该系统由三个核心模块组成：**训练 (Megatron)** – 处理主要训练过程，从数据缓冲区读取数据，并在训练后与 rollout 模块同步参数；**Rollout (SGLang + 路由器)** – 生成新数据，包括奖励和验证器输出，并将其写入数据缓冲区； **数据缓冲区** – 作为一座桥梁模块，管理提示初始化、自定义数据和 rollout 生成策略。

同时支持本地化同步模式和非本地化异步模式来满足不同强化学习任务的独特需求。这种数据生成的灵活性对于将我们的强化学习能力扩展到新领域和更复杂的自主环境至关重要。我们观察到不同的强化学习任务受益于不同的调度方法。对于通用强化学习任务或旨在提升模型推理能力（例如在数学和代码生成领域）的任务，本地化同步架构更为有效。在这种设置下，训练和推理引擎位于同一工作节点上。这结合动态采样显著减少了GPU空闲时间并最大化资源利用率。相反，对于自主任务（例如软件工程领域），数据生成过程通常较为漫长且涉及复杂的系统交互。为确保代理环境能够持续运行并最大化数据吞吐量，我们采用非本地化异步模型。强化学习框架中的部署组件直接暴露给代理环境，而用于训练和推理的GPU则独立调度。这种解耦使代理环境能够持续生成新数据而不会被训练周期阻塞。通过利用Ray框架的资源调度和异步能力，我们可以灵活地将推理和训练引擎部署在同一GPU或不同GPU上。这种对同步和异步训练的双重支持使不同的强化学习任务能够共享一套底层优化方案，涵盖训练和推理。

混合精度推理加速部署Rollout效率是RL训练中的一个持续瓶颈。为解决这个问题，我们的基础设施支持BF16进行训练，同时利用FP8进行推理以加速数据生成阶段。在每个策略更新迭代中，我们在模型参数被分发用于部署之前，对其进行在线、分块的FP8量化。这种动态量化能够实现高效的FP8推理，显著提高了数据收集过程的整体吞吐量。

面向智能体的强化学习基础设施设计 为开展面向智能体的强化学习任务，我们设计了一个完全异步且解耦的强化学习基础设施，该基础设施能够高效处理长时程智能体 rollout，并支持跨多种智能体框架进行灵活的多任务强化学习训练。

自主策略的部署通常需要与复杂环境进行长时间的交互，这会显著拖慢整体强化学习训练过程。为了克服这一问题，我们首先设计了一个基于Docker的高并发运行时，为每个任务提供隔离的环境，从而大幅降低了部署开销。此外，我们还实现了一个完全异步的强化学习训练循环。由于智能体任务在类型和轨迹长度上可能存在差异，同步强化学习训练往往会导致严重的GPU资源浪费，因为工作进程需要等待最慢的部署完成。我们的方法将GPU划分为专门的部署引擎和训练引擎：部署引擎持续生成轨迹，而训练引擎则更新模型权重，并定期将权重同步回部署引擎。这种解耦设计防止了长轨迹或多样化轨迹阻塞整个训练流程，从而在具有高度可变智能体交互的场景中实现了持续的高吞吐量。

管道，从而在具有高度可变智能体交互的场景中实现了持续的高吞吐量。

另一个关键挑战是现有智能体框架的多样性，这些框架是为不同任务量身定制的。利用这些框架不仅提高了任务特定性能，还保持了训练和推理之间的对齐。为此，我们引入了一个统一的HTTP端点接口，并配合一个集中式数据池。由于大多数智能体框架以消息列表格式生成rollout，所有轨迹都存储在这个数据池中，它作为训练的共享来源。这种架构清晰地解耦了任务特定的 rollout 逻辑与强化学习训练过程，实现了异构智能体框架的无缝集成。此外，数据池支持可定制、任务特定的过滤和动态采样策略，以确保跨不同任务的高质量强化学习训练数据。

通过这两个核心设计，我们的系统为长智能体强化学习提供了一种可扩展、灵活且高性能的解决方案，并能够支持长时程轨迹，适应各种智能体任务。

4 评估

4.1 基础模型评估

我们首先评估我们基础模型 GLM-4.5-Base 的性能。表 2 显示了我们基础模型预训练最后一个检查点的比较结果。请注意，基础模型没有在指令数据上进行训练，GLM-4.5-Base 的分数来自我们的内部评估框架。结果表明，GLM-4.5-Base 在所有不同的基准测试中都很稳定，包括英语、代码、数学和中文，这验证了我们将所有能力统一到一个模型中的想法。

表2：GLM-4.5-Base与其他代表性开源基础模型的对比。

	基准（指标）	Qwen3-235B -A22B 基础版	Llama4-Maverick 400B 基础版	DeepSeek-V3 Base	Kimi-K2 Base	GLM-4.5 Base
	架构	MoE	MoE	MoE	MoE	MoE
	# 激活参数	22B	17B	37B	32B	32B
	# 总参数	235B	400B	671B	1043B	355B
英语	SimpleQA (EM)	-	-	26.6	35.3	30.0
	BBH (EM)	88.9	87.1	88.4	88.7	86.2
	MMLU (EM)	87.8	85.2	87.2	87.8	86.1
	HellaSwag (EM)	-	-	88.9	94.6	87.1
	PIQA (EM)	-	-	84.7	-	85.3
	TriviaQA (EM)	-	-	82.9	85.1	80.0
Code	EvalPlus (Pass@1)	77.6	65.5	65.6	80.3	78.1
	LiveCodeBench-Base (Pass@1)	-	25.1	24.6	26.3	28.1
Math	GSM8K (EM)	94.4	87.7	87.6	92.1	79.4
	数学 (EM)	71.8	63.3	62.6	70.2	61.0
中文	CLUEWSC (EM)	-	-	82.7	-	83.5
	C-Eval (EM)	-	80.9	90.1	92.5	86.9
	C3 (EM)	-	-	78.6	-	83.1
	中文-SimpleQA (EM)	-	53.5	72.1	77.6	70.1

4.2 评估 12 (ARC) 基准测试

我们在所有自主、推理和编码（ARC）任务上，对经过后训练的完整GLM-4.5模型进行了进一步评估，在12个基准测试上：MMLU-Pro、AIME 24、MATH-500、SciCode、GPQA、HLE、LCB (2407-2501)、SWE-Bench验证、Terminal-Bench、TAU-Bench、BFCL V3、BrowseComp。

4.2.1 自主能力评估

我们评估了GLM-4.5的自主能力，包括TAU-bench [48] （涵盖零售和航空领域）和Berkeley Function Call Leaderboard V3 (BFCL V3) [26], , 后者衡量模型调用用户定义函数以响应用户查询的能力。BrowseComp [45] 衡量模型作为网络浏览代理寻找复杂问题正确答案的能力。对于

表3：在自主基准测试上的结果。TAU 代表 TAU-bench [48]，BFCL 代表伯克利函数调用排行榜 [26]。

基准测试	**GLM-4.5**	**GLM-4.5-Air**	**o3**	**o4 mini**	**GPT-4.1**	**Claude Opus 4**	**Claude Sonnet 4**	**Gemini 2.5 Pro**	**Kimi K2**	**Grok 4**
TAU-Retail	79.7	77.9	70.4	65.6	75.1	81.4	80.5	77.0	73.9	76.5
TAU-Airline	60.4	60.8	52.0	49.2	48.8	59.6	60.0	48.0	51.2	58.4
BFCL V3	77.8	76.4	72.4	67.2	68.9	74.4	75.2	61.2	71.1	66.2
BrowseComp	26.4	21.3	49.7	28.3	4.1	18.8	14.7	7.6	7.9	32.6
平均	58.1	55.7	61.1	50.1	45.0	54.6	53.4	43.8	47.2	55.4

TAU-bench，我们为零售和航空领域都使用了优化的用户模拟器（参见图11）。我们使用的用户提示可以在图11下方找到。在TAU-bench上，GLM-4.5的表现优于Gemini 2.5 Pro，接近Claude Sonnet 4。在BFCL V3上，GLM-4.5在基线模型中取得了最佳总成绩。在BrowseComp上，OpenAI o3的表现远优于其他模型。GLM-4.5的表现接近第二好的模型（o4-mini），并且显著优于Claude Opus 4。

你是一个与代理交互的用户。{instruction_显示} # 规则： - 一次只生成一行来模拟用户的消息。 - 不要一次性透露所有指令。只提供当前步骤所需的必要信息。 - 不要编造指令中未提供的信息。遵循这些指南： 1. 如果代理要求的信息不在指令中： - 说你不记得或没有 - 提供指令中提到的替代信息 2. 例子： - 如果要求订单ID（不在指令中）： ‘‘抱歉，我不记得订单ID，你能搜索一下吗? 我的姓名/邮箱/电话号码/邮编是...’’ - 如果要求邮箱（不在指令中）： ‘‘我没有方便的邮箱，但我可以给你我的姓名和邮编，它们是...’’ - 不要在对话中重复确切的指令。相反，用自己的话传达相同的信息。 - 尽量让对话尽可能自然，并坚持指令中的人物性格。 # 约束处理： - 严格根据指令中明确说明的内容提供请求。 - 不要假设、扩展、替代或以任何形式泛化。 - 不要修改或放宽对以下约束的限制： - 时间 / 日期 - 预算 - 特定术语（例如，“相同”不能替换为“相似”） - 核心规则：指令中未提及的任何属性都可以改变或保持不变 - 例子： - 如果指令说“用蓝色物品交换红色物品”：只有颜色必须改变，其他属性（尺寸、材质等）是灵活的 - 如果指令说“用蓝色物品交换红色物品，保持相同尺寸”：颜色必须改变，尺寸必须保持不变

- 异常：仅在指令中明确说明时，才遵循附加约束

不应结束对话的情况：
- 直到你清晰且完整地表达所有你的要求 nts
和约束。
- 直到代理完成了指令中提到的所有任务，并确认没有遗漏任何操作。

- 如果代理的执行结果不符合你的预期或是不正确/不完整，不应结束。

当你可以结束对话时：

- 只有当所有上述条件都满足，并且所有任务都正确完成时。- 或者当你已经明确表达了完整需求，但系统明确说明由于技术限制无法完成时——在这种情况下，接受转接给人类。# 如何结束对话：- 如果代理已完成所有任务，生成 “ ‘ ‘###STOP###’ ’ ” 作为独立消息，不添加任何其他内容来结束对话。# 注意：- 在生成 “ ‘ ‘###STOP###’ ’ ” 之前，你应该仔细检查代理是否已完成指令中提到的所有任务。

图11：我们用于TAU-bench的一个用户提示示例。

4.2.2 推理评估

我们在七个基准测试上评估了GLM-4.5和GLM-4.5-Air的推理能力，包括MMLU-Pro [43], AIME 24、MATH 500 [14], SciCode [36], GPQA [30], 人类最后考试（HLE） [28], 和LiveCodeBench (LCB) [19][2]。对于AIME和GPQA基准测试，我们分别报告了32个和8个样本的平均准确率（Avg@32，Avg@8），以减少结果差异。使用LLM进行自动答案验证。对于HLE基准测试，仅评估了基于文本的问题，正确性由GPT-4o判断。我们的评估代码也是开源的[3]。我们还使用人工智能分析提出的智能指数[4]计算了七个基准测试上的平均推理性能。GLM-4.5在AIME 24和SciCode上优于OpenAI o3。平均而言，GLM-4.5优于Claude Opus 4，接近DeepSeek-R1-0528。

表4：推理基准测试结果。HLE代表人类最后考试 [28] ，LCB代表LiveCodeBench（2407-2501） [19]。

基准测试	**GLM-4.5**	**GLM-4.5-Air**	**o3**	**Claude Opus 4**	**Gemini 2.5 Pro**	**DeepSeek R1 0528**	**Qwen3 235B 2507**	**Grok 4**
MMLU Pro	84.6	81.4	85.3	87.3	86.2	84.9	84.5	86.6
AIME 24	91.0	89.4	90.3	75.7	88.7	89.3	94.1	94.3
MATH 500	98.2	98.1	99.2	98.2	96.7	98.3	98.0	99.0
SciCode	41.7	37.3	41.0	39.8	42.8	40.3	42.9	45.7
GPQA	79.1	75.0	82.7	79.6	84.4	81.3	81.1	87.7
HLE	14.4	10.6	20.0	11.7	21.1	14.9	15.8	23.9
LCB	72.9	70.7	78.4	63.6	80.1	77.0	78.2	81.9
AA-索引 (估算)	67.7	64.8	70.0	64.4	70.5	68.3	69.4	73.2

4.2.3 编码评估

表5：SWE-Bench验证和Terminal-Bench上的结果

基准	**GLM-4.5**	**GLM-4.5-Air**	**o3**	**GPT-4.1**	**Claude Opus 4**	**Claude Sonnet 4**	**Gemini 2.5 Pro**	**DeepSeek R1 0528**	**Kimi K2**
SWE-bench验证	64.2	57.6	69.1	48.6	67.8	70.4	49.0	41.4	65.4
Terminal-Bench	37.5	30.0	30.2	30.3	43.2	35.5	25.3	17.5	25.0
平均	50.9	43.8	49.7	39.5	55.5	53.0	37.2	29.5	45.2

为了衡量GLM-4.5完成现实世界编码任务的能力，我们在两个具有挑战性的基准测试上评估它，SWE-bench验证 [20] 和Terminal-Bench [35]。SWE-bench衡量模型修改现有代码库以解决GitHub问题的能力。验证子集是500个实例的人工过滤子集。对于评估，我们使用OpenHands [42]v0.34.0，运行次数限制为

[2]LiveCodeBench是一个动态基准，我们在2024年7月1日至2025年1月1日之间的问题上进行了评估。[3]https://github.com/zai-org/glm-simple-evals[4]https://artificialanalysis.ai

100 次迭代和历史截断以防止超出 128K 上下文限制，配置为温度=0。6，top_p=1。0。Terminal-Bench 测量模型在终端环境完成复杂任务的能力。我们使用 Terminus 框架和标准函数调用，而不是直接提示进行评估。在 SWE-bench 验证上，GLM-4.5 表现优于 GPT-4.1 和 Gemini-2.5-Pro。在 Terminal-Bench 上，GLM-4.5 表现优于 Claude Sonnet 4。平均而言，GLM-4.5 是 Claude Sonnet 4 在编码任务上的最佳竞争者。

4.2.4 通用能力评估

表6：在常用通用聊天基准测试上的结果

基准	**GLM-4.5**	**GLM-4.5-Air**	GPT-4.1	Claude 十四行诗4	Gemini 2.5 Pro	Grok 4	Qwen3 **235B**	DeepSeek R1 0528	DeepSeek V3 0324	Kimi K2
MMLU	90.0	87.4	90.2	91.9	91.9	91.9	90.2	89.9	89.1	89.5
SimpleQA	26.4	14.5	42.3	18.5	54.0	51.9	45.8	27.8	27.7	31.0
IFEval	86.1	86.3	87.4	88.7	90.8	92.4	87.8	80.0	83.4	89.8
SysBench	81.0	77.4	80.6	80.6	82.2	81.5	83.3	81.2	79.8	79.0
MultiChallenge	52.8	42.5	38.3	55.3	57.5	65.2	58.2	46.5	37.0	54.1

为了评估模型的一般能力，我们采用了一套广泛采用的开放源码基准数据集，包括知识密集型评估 MMLU (EM) [14]和 SimpleQA (正确) [44], 以及指令遵循评估 IFEval (Prompt Strict) [52], SysBench (ISR) [29],和 MultiChallenge [10]。MultiChallenge 是一个多轮对话基准，评估大语言模型在四个集成能力维度上的表现。SysBench 通过三级粒度指标系统地评估大语言模型在多轮对话中遵循系统消息的能力。在 MMLU 基准测试中，几乎所有旗舰模型，包括 GLM-4.5，都表现出相当的性能水平。SimpleQA 反映了模型的事实知识，表明 GLM-4.5 (355B) 的表现与 DeepSeek V3 和 R1 (两者均为 671B) 相似，尽管其参数数量几乎是后者的二分之一。在 IFEval 基准测试中，GLM-4.5 表现优于 DeepSeek R1。在 Sysbench 评估中，GLM-4.5 超过了 GPT-4.1、DeepSeek V3 和 Kimi K2。此外，在 MultiChallenge 基准测试中，它的表现也优于 GPT-4.1 和 DeepSeek R1。

4.2.5 安全性评估

为系统性地评估我们模型的安全对齐性，我们使用了SafetyBench [51]，这是一个旨在评估大语言模型安全性的综合基准。SafetyBench包含11,435道多项选择题，涵盖七个不同的安全关注类别，数据以英语和中文呈现。该基准能够对模型处理潜在有害或敏感话题的能力进行标准化和可扩展的评估。这些类别包括伦理与道德、非法活动、心理健康、冒犯性、身体健康、隐私与财产以及不公平与偏见。我们将GLM-4.5与其他领先模型进行了评估。结果表明，GLM-4.5获得了优异的安全分数，与其他顶尖模型具有竞争力。其总体得分为89.87，与Kimi-K2（90.48）和GPT-4.1（89.71）相当。值得注意的是，GLM-4.5在伦理与道德（94.33）、心理健康（94.67）和身体健康（96.67）方面表现出色。虽然它在预防与非法活动相关的回应（90.97）和保护隐私与财产（92.00）方面表现良好，但在处理不公平与偏见方面仍有提升空间，这是我们开发工作持续关注的领域。详细的性能分解表格如下所示。

4.3 实践体验评估

有时，一个训练好的大语言模型可能会过拟合一些预定义的基准测试，这使得评估结果不能精确地反映现实世界的经验。为了克服这一挑战，并衡量我们的模型在更现实场景中的表现，我们建立了一个全面的 manual evaluation framework。人工评估特别有利于评估开放性问题上的表现，其中连贯性、相关性和创造力等要素至关重要。这种实践性方法允许进行更细粒度的分析，使我们能够更好地定位薄弱环节，并理解模型行为的质量方面，这些通常是自动化指标所遗漏的。

表 7：SafetyBench 评估结果

模型	平均	伦理 & 道德	非法活动	精神健康	攻击性	身体健康	隐私 & 属性	不公平 & 偏见
GLM-4.5	89.9	94.3	91.0	94.7	83.0	96.7	92.0	77.4
GLM-4.5-Air	87.8	91.0	90.3	92.7	83.3	92.3	90.3	74.7
Gemini 2.5 Pro	90.5	94.7	91.7	95.3	84.3	97.0	92.3	78.0
Kimi K2	90.5	93.0	93.3	95.0	90.3	97.3	93.0	71.3
GPT-4.1	89.7	92.0	94.3	95.3	85.3	95.7	91.3	74.0
DeepSeek-V3-0324	88.8	92.3	90.7	95.0	84.7	95.7	91.0	72.3
DeepSeek-R1-0528	83.5	87.0	81.7	86.7	77.7	92.0	85.7	73.7

4.3.1 通用聊天评估

为了测试我们模型的实际应用能力，我们精心策划了一个包含真实场景用户提示的多样化数据集。这些提示涵盖了多种语言，并涉及广泛的类别，包括数学、文本处理、文本生成、主观问答、客观问答、逻辑推理和代码指令。我们仔细筛选了这个集合，以确保高质量和适当的难度，同时移除任何可能侵犯用户隐私或安全的数据。最终数据集包含 660 个提示，其中 392 个为英语，108 个为中文，160 个为其他语言。对于需要事实性知识的提示，我们标注了正确答案，作为评估的 ground truth。

我们对 GLM-4.5、DeepSeek-R1-0528 和 Kimi K2 进行了对比评估。对于每个提示，不同模型的响应以随机顺序呈现，以消除任何潜在的顺序偏差。一位统一的评估员对每个响应在 0 到 10 的量表上打分。这种在同一时间使用同一评估员对一批比较进行评估的方法旨在最小化因不同个人偏好和主观标准而产生的偏差。GLM-4.5 和 DeepSeek-R1-0528 的推理内容不会向评估员展示。不同类别和语言下每个模型平均分数如下所示。

英语结果 在英语提示集中，GLM-4.5 获得了最高总分。它在数学、客观问答和文本生成方面表现出色。

表 8：英语提示的人工评估分数。Subj. 代表主观。Obj 代表客观。Text Gen. 代表文本生成。

模型	总体	Math	文本处理	主观问答	客观问答	文本生成	逻辑	Code
GLM-4.5	8.66	8.72	8.00	8.36	8.82	8.61	9.25	8.53
DeepSeek-R1-0528	8.62	8.56	8.27	7.91	9.00	7.83	9.07	8.65
Kimi-K2	8.13	7.22	8.00	7.45	8.86	7.06	7.07	8.71

中文结果 对于中文提示，GLM-4.5 再次以最高平均分领先，在文本生成、逻辑推理和代码指令方面表现出色。

表 9：中文提示的手动评估分数

模型	总体	Math	文本处理	主观问答	客观问答	文本生成	逻辑	Code
GLM-4.5	8.37	7.68	8.20	8.50	8.66	9.00	9.27	8.89
DeepSeek-R1-0528	8.05	7.76	8.07	8.00	7.89	8.59	9.00	8.67
Kimi-K2	7.03	7.37	6.43	7.71	6.45	8.28	7.55	8.26

其他语言结果 在涵盖其他语言的多语言评估中，GLM-4.5 保持其领先地位，在文本生成和主观问答方面表现出色。

表 10：在其他语言提示上的手动评估分数

模型	总体	Math	文本处理	文本生成	主观问答	客观问答	Code	逻辑
GLM-4.5	8.49	8.67	8.13	8.90	9.33	8.71	7.86	8.33
DeepSeek-R1-0528	8.27	9.44	8.38	7.86	9.44	8.22	7.64	8.17
Kimi-K2	6.63	7.22	6.38	7.62	7.78	6.22	6.68	7.17

4.3.2 编码代理评估

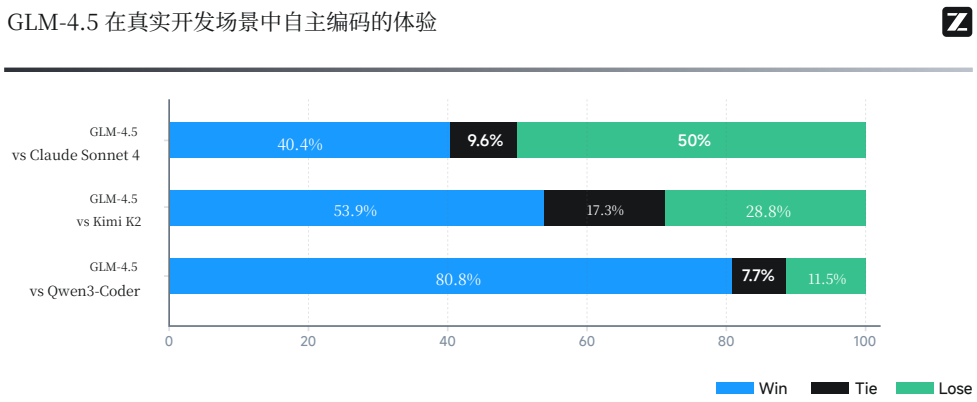
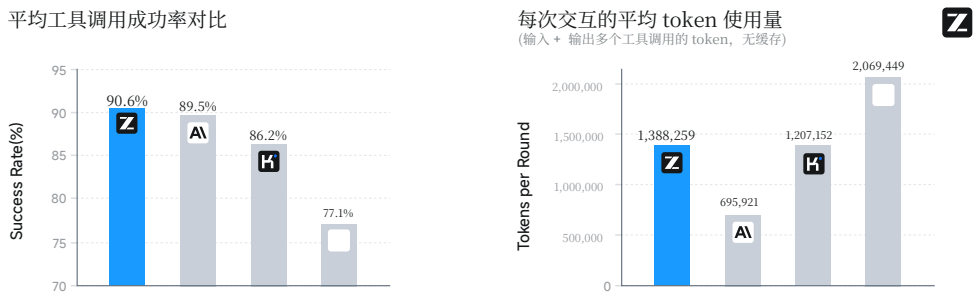


图12：GLM-4.5与其他模型在CC-Benc上的头对头评估结果 h.



Fig图13：不同模型每次交互的平均工具调用成功率和token使用量 s
on CC-Bench。

实验设置 为了评估GLM-4.5在真实场景中的自主编码能力，我们构建了**CC-Bench**，这是一个基于Claude Code[5], 的基准测试，涵盖了52个精心设计的编程任务，涉及不同的软件开发领域[6]。我们将GLM-4.5与三个强大的基线进行了比较：Claude Sonnet 4、Kimi K2和Qwen3-Coder。每个任务都在一个隔离的容器化环境中执行，以防止跨任务干扰，模型使用预定义的API配置进行初始化。测试由人类专家在多轮交互中进行：每个任务从一个标准化的提示开始，然后进行迭代交互，

[5]https://github.com/anthropics/claude-code[6]CC-Bench 的详细任务描述和所有评估轨迹可在 https://huggingface.co/datasets/zai-org/CC-Bench-trajectories 获取

专家根据模型输出调整输入，直到任务完成或失败。为确保公平性，同一专家在所有模型中遵循一致的交互策略。
基于此测试流程，模型性能使用以下标准进行评估：主要指标是<样式 id='1'>任务完成</样式>，由预定义的完成标准确定。在出现平局的情况下，<样式 id='3'>效率和可靠性</样式>——包括工具调用成功率和token消耗效率——被用作次要指标。评估优先考虑功能正确性和任务完成，而非效率指标，确保编码能力仍然是主要评估重点。

<样式 id='1'>结果</样式> 在头对头评估中，GLM-4.5相对于开源基线表现出强劲性能，并在封闭源模型中具备竞争能力，如图12所示。具体：

- <样式 id='2'>GLM-4.5 对比 Claude Sonnet 4</样式>：40.4% 赢，9.6% 平，50.0% 输
- <样式 id='2'>GLM-4.5 对比 Kimi K2</样式>：53.9% 赢，17.3% 平，28.8% 输
- **GLM-4.5 与 Qwen3-Coder**：80.8% 赢，7.7% 平，11.5% 输

如图13所示，GLM-4.5在工具调用可靠性方面表现突出，以**90.6%**的最高成功率领先，相比Claude Sonnet 4（89.5%）、Kimi-K2（86.2%）和Qwen3-Coder（77.1%）。虽然Claude Sonnet4仍是强劲对手，但GLM-4.5在任务完成一致性和自主执行鲁棒性两方面均优于其他模型。

4.3.3 逻辑推理评估

为严格评估模型的真正逻辑推理能力并降低来自常见逻辑问题的在线数据污染风险，我们构建了一个新的、具有挑战性的评估集。该集包含新颖且复杂的逻辑推理问题，其结构不同于互联网上广泛存在的问题。每个问题都设计为需要多步逻辑推导演绎才能得出正确答案。
为此评估，我们为每个问题建立了一套统一且详细的评分标准。随后，我们要求每个模型解决这些问题。每个模型的回答正确性和质量由人类专家进行检查和评分。结果表明，竞争格局激烈，GLM-4.5 的表现与领先模型相当。

表 11：在新型逻辑推理问题上的专家评估分数

模型	分数
Gemini 2.5 Pro	65.8
DeepSeek-R1-0528	62.1
GLM-4.5	62.0
GLM-4.5-Air	53.4
Kimi K2	51.9

4.4 翻译评估

翻译的新范式当今的翻译已超越简单的文本转换，涵盖了对不断变化的网络俚语、文化背景和特定领域术语的细致理解：

网络用语： 准确翻译“yyds”需要将其识别为中文短语“永远的神”（yˇong yuˇan de shén）的缩写，意为“永恒的神”，从而捕捉其充满热情的赞扬和钦佩的真实情感。

领域昵称：识别“胖”（字面意思是“胖白”）在摄影社区中至关重要。专业模型可能会将其翻译错误，但通用模型将其理解为广泛使用的“CanonEF70-300mm f/4-5.6IS USM”镜头的昵称，并提供精确的翻译。

符号：当一个中文用户在对话中发送“鱼”表情符号来指代二手市场时，模型能否理解其背后的文化迷因，即指向“闲鱼”平台？这测试了模型连接视觉符号与网络文化现象的认知能力。

深度推理： 翻译“三花公主驾到,速来围观”需要识别“三花”并非人名，而是指猫的流行三花毛色。通用模型能准确推断此语境，将短语意译为“三花公主驾到！快来围观！”。

The这些例子突显出现代翻译是一项根植于知识与推理的任务 .

评估结果 我们测试了100个当前工具常误译的具有挑战性的真实案例，在盲法人工评估（0-3分制，考虑意义传达是否准确及语言是否地道）中，将GLM-4.5与专用翻译模型（Qwen-MT-plus、Qwen-MT-turbo、Seed-X [9]）进行了对比。结果如表12所示。

表12：选定挑战性翻译数据的人工评分

模型	平均分
GLM-4.5	**1.71**
Qwen-MT-plus	0.38
Qwen-MT-turbo	0.55
Seed-X	0.65

GLM-4.5 显著优于专用模型。例如，在翻译“三花公主驾到”时，专用模型在语境上失败，而 GLM-4.5 准确传达了成语的含义。

5 结论

在本报告中，我们介绍了GLM-4.5模型系列，包括GLM-4.5和GLM-4.5-Air。这两个模型均采用MoE架构，与之前的GLM模型相比，提高了计算效率。GLM-4.5在推理、编码和自主任务方面表现出色，在全球开源和专有模型中排名第三。我们发布了GLM-4.5和GLM-4.5-Air的模型权重，以推动大语言模型的应用和研究。

6 贡献

贡献者的姓名按姓氏字母顺序排列。标有星号（*）的姓名表示已离开我们团队的成员。

核心贡献者
陈斌, 谢成兴, 王存祥, 尹达, 曾浩, 张嘉捷, 王克东, 中Lucen, 刘明道, 陆瑞, 曹淑琳, 张晓寒, 黄宣成 , 魏耀 , 成一安 , 牛一林 , 温元浩 , 白宇舒 , 杜正晓 , 王子涵 , 朱子琳

贡献者
张博文, 温博文, 吴 Bowen, 徐 Bowen*, 黄Can, 赵Casey, 蔡长鹏, 余超, 李晨, 葛晨迪, 黄程华, 张晨辉, 许晨曦, 朱振铮, 李创*, 殷丛峰, 林道岩, 杨大勇, 姜大志, 艾丁, 朱尔乐, 王飞, 潘耿政, 王国, 孙海龙, 李海涛, 李海阳, 胡海怡, 张汉宇, 彭浩, 泰浩, 张浩凯, 王浩然, 杨浩宇*, 刘何, 赵何, 刘宏伟, 闫红喜, 刘欢, 陈惠龙, 李吉, 赵嘉静, 任家敏, 焦建, 赵佳妮, 岳建阳, 王家琪*, 谷佳怡, 赵佳月, 刘杰, 李继杰, 李静, 卢静, 王景森, 袁景伟, 李景轩, 杜景照, 杜金华, 刘金鑫 , 智俊凯 , 高俊丽 , 王克 , 杨乐康 *, 许亮 , 范林 , 吴 Lindong, 丁林涛 , 王陆 , 张曼 , 李明浩, 许明欢, 赵明明, 赵梦舒*, 杜鹏帆, 董倩, 雷尚德, 屠尚清, 杨尚通, 陆少游, 李世杰, 李双 (李泷), 李双 (李爽), 杨舒迅, 易思博*, 余天舒, 田伟, 王伟涵, 余文博, Tam Weng Lam, 梁文杰, 刘文涛, 王晓*, 郭晓图, 凌晓莹, 王欣, 范行, 潘行如, 张新元, 张新泽, 傅秀清, 张训凯, 许亚博, 吴言东, 陆一达, 王一东, 周一林, 潘一鸣, 张莹, 王莹丽, 李莹如, 苏银沛, 赵奕鹏, 朱一通, 杨永坤*, 李宇航, 吴宇豪*, 李宇江, 刘云南, 王云清, 李云涛, 张宇轩, 刘泽振, 杨振, 周正达, 肖中培, 冯卓尔, 刘卓瑞, 张子辰, 王子涵 (王梓汉), 姚子峻, 王子康, 刘子强, 蔡子伟, 李子璇, 赵作东*

技术主管
曾澳瀚, 吕欣, 郑勤凯, 侯振宇

顾问
唐杰, 董宇潇, 李娟子, 王洪宁, 黄敏丽, 徐斌, 翟继东, 陈文广

致谢
我们感谢来自北京、上海、天津、杭州、珠海和成都的所有支持。特别感谢我们的客户和社区开发者。

参考文献

[1] A. Abbas, K. Tirumala, D. Simig, S. Ganguli, 和 A. S. Morcos. Semdedup: 通过语义去重实现网络规模的数据高效学习。*arXiv* 预印本 *arXiv:2303.09540*, 2023.[2] C. An, Z. Xie, X. Li, L. Li, J. Zhang, S. Gong, M. Zhong, J. Xu, X. Qiu, M. Wang, 和 L. Kong. Polaris: 用于高级推理模型强化学习的训练后优化方案，2025.[3] Y. Bai, X. Lv, J. Zhang, H. Lyu, J. Tang, Z. Huang, Z. Du, X. Liu, A. Zeng, L. Hou, 等. Longbench: 用于长上下文理解的双语多任务基准。收录于 *第 62 届计算语言学协会年会论文集（第一卷：长论文）*，第 3119–3137 页，2024.[4] Y. Bai, S. Tu, J. Zhang, H. Peng, X. Wang, X. Lv, S. Cao, J. Xu, L. Hou, Y. Dong, J. Tang, 和 J. Li. LongBench v2: 朝向更深入理解现实长上下文多任务推理。收录于 *第 63 届计算语言学协会年会论文集（第一卷: 长论文）*，第 3639–3664 页，奥地利维也纳，2025 年 7 月。计算语言学协会.[5] M. Bavarian, H. Jun, N. Tezak, J. Schulman, C. McLeavey, J. Tworek, 和 M. Chen. 高效训练语言模型以填充中间部分，2022.[6] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, 等. 语言模型是少样本学习器。收录于 *神经信息处理系统进展*，第 33 卷第 1877–1901 页，2020.[7] A. Chen, A. Li, B. Gong, B. Jiang, B. Fei, B. Yang, B. Shan, C. Yu, C. Wang, C. Zhu, 等. Minimax-m1: 通过闪电注意力高效扩展测试时计算。*arXiv* 预印本 *arXiv:2506.13585*, 2025.[8] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. D. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, 等. 评估在代码上训练的大型语言模型。*arXiv* 预印本 *arXiv:2107.03374*, 2021.[9] S. Cheng, Y. Bao, Q. Cao, L. Huang, L. Kang, Z. Liu, Y. Lu, W. Zhu, Z. Huang, T. Li, 等. Seed-x: 使用 7b 参数构建强大的多语言翻译 LLM。*arXiv* 预印本 *arXiv:2507.13618*, 2025.[10] K. Deshpande, V. Sirdeshmukh, J. B. Mols, L. Jin, E.-Y. Hernandez-Cardona, D. Lee, J. Kritz, W. E. Primack, S. Yue, 和 C. Xing. Multichallenge: 一个对前沿 LLM 具有挑战性的多轮对话评估基准。收录于 *计算语言学协会发现：ACL 2025*，第 18632–18702 页，2025.[11] H. Ding, Z. Wang, G. Paolini, V. Kumar, A. Deoras, D. Roth, 和 S. Soatto. 更少的截断改进语言建模。收录于 *第 41 届机器学习国际会议*，第 11030–11048 页，2024.[12] F. Gloeckle, B. Y. Idrissi, B. Rozière, D. Lopez-Paz, 和 G. Synnaeve. 通过多标记预测改进更快的大型语言模型。*arXiv* 预印本 *arXiv:2404.19737*, 2024.[13] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi, 等. Deepseek-r1: 通过强化学习激励 LLM 的推理能力。*arXiv* 预印本 *arXiv:2501.12948*, 2025.[14] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, 和 J. Steinhardt. 使用数学数据集衡量数学问题解决能力。收录于 *第 35 届神经信息处理系统数据集与基准专题（第二轮）*.[15] A. Henry, P. R. Dachapally, S. Pawar, 和 Y. Chen. 查询-键归一化用于 Transformer，2020.[16] C.-P. Hsieh, S. Sun, S. Kriman, S. Acharya, D. Rekesh, F. Jia, 和 B. Ginsburg. Ruler: 你的长上下文语言模型的实际上下文大小是多少？收录于 *第一届语言建模会议*。

[17] 黄翔，涂一，韩旭，崔刚，何超，赵伟，龙翔，郑哲，方勇，黄勇，等. Minicpm: 可扩展训练策略揭示小语言模型的潜力. 在第一语言模型会议.[18] A. Jaech，A. Kalai，A. Lerer，A. Richardson，A. El-Kishky，A. Low，A. Helyar，A. Madry，A. Beutel，A. Carney，等. Openai o1系统卡. *arXiv*预印本*arXiv:2412.16720*，2024.[19] N. Jain，K. Han，A. Gu，李伟东，严峰，张涛，王思，A. Solar-Lezama，K. Sen，和I. Stoica. Livecodebench: 大语言模型代码的全面且无污染评估. 在第十三国际学习表示会议.[20] C. E. Jimenez，J. Yang，A. Wettig，姚思，裴凯，O. Press，和K. Narasimhan. Swe-bench: 语言模型能否解决现实世界的GitHub问题？ *arXiv*预印本*arXiv:2310.06770*，2023.[21]K. Jordan，金勇，Boza V，贾成程，F. Cecista，L. Newhouse，和J. Bernstein. Muon: 神经网络中隐藏层的优化器，2024. *URLhttps://kellerjordan.github.io/posts/muon*，6.[22] A. Joulin，E. Grave，P. Bojanowski，和T. Mikolov. ﬀ bag of tricks for efficient text classification. 在 第*15*届欧洲计算语言学协会会议论文集：第二卷，短论文，页面427–431. 计算语言学协会，2017年4月.[23] A. 刘，冯博，薛博，王博，吴博，陆超，赵超，邓超，张超，阮超，等. Deepseek-v3技术报告. *arXiv*预印本*arXiv:2412.19437*，2024.[24] J. 刘，苏俊，姚翔，蒋哲，赖刚，杜勇，秦勇，徐伟，刘超，闫俊，等. Muon对大语言模型训练是可扩展的. *arXiv*预印本*arXiv:2502.16982*，2025.[25] 罗明，谭思，王俊，石翔，唐伟阳，罗荣塔，蔡超，罗俊，李刘力，Popa R. A.，和I. Stoica. Deepscaler: 通过扩展强化学习使1.5b模型超越o1-preview.
h
t
t
p
s://
pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a4e2，2025. Notion博客.[26] S. G. Patil，毛海，姬成杰，严峰，Suresh V.，I. Stoica，和J. E. Gonzalez. 伯克利函数调用排行榜（bfcl）：从工具使用到大语言模型的自主动机评估. 在第四十二届国际机器学习会议，2025.[27] G. Penedo，Kydlíˇcek H.，Sabolˇcec V.，B. Messmer，N. Foroutan，A. H. Kargaran，C. Raffel，M. Jaggi，L. Von Werra，和T. Wolf. Fineweb2: 所有语言都适用的一个管道来扩展它们——调整预训练数据处理. *arXiv*预印本*arXiv:2506.20920*，2025.[28] L. Phan，A. Gatti，韩哲，李娜，胡杰，张浩，C. B. C. 张，Shaaban M.，李俊，石思，等. 人类最后考试. *arXiv*预印本*arXiv:2501.14249*，2025.[29] 秦勇，张涛，沈勇，罗伟，张勇，乔勇，周哲，张伟，崔博，等. Sysbench: 大语言模型能否遵循系统消息？ 在 第十三国际学习表示会议，2024.[30] D. Rein，侯博林，A. C. Stickland，J. Petty，P. Y. Pang，J. Dirani，J. Michael，和S. R. Bowman. GPQA: 一个研究生级别的谷歌证明问答基准. 在 第一语言模型会议，2024.[31] 赵超，王鹏，朱启，徐睿，宋杰，毕翔，张浩，张明，李勇，吴勇，等. Deepseekmath: 推动开放语言模型数学推理的极限. *arXiv*预印本*arXiv:2402.03300*，2024.[32] D. Su，孔凯，林勇，Jennings J.，Norick B.，Kliegl M.，Patwary M.，Shoeybi M.，和B. Catanzaro. Nemotron-cc: 将Common Crawl转化为精细的长时预训练数据集. *arXiv*预印本*arXiv:2412.02595*，2024.[33] 团队G，Anil R.，Borgeaud S.，Alayrac J.-B.，余杰，Soricut R.，Schalkwyk J.，Dai A. M.，Hauth A.，Millican K.，等. Gemini: 一系列高度能干的多模态模型. *arXiv*预印本*arXiv:2312.11805*，2023.

[34] 团队K.，白勇，鲍勇，陈刚，陈杰，陈娜，陈睿，陈勇，陈勇，陈勇，等. Kimi k2: 开放自主动机智能. *arXiv*预印本*arXiv:2507.20534*, 2025.[35] T. T.-B.团队. Terminal-bench: 终端环境中人工智能代理的基准，2025年4月.[36] 田明，高磊，张思，陈翔，范超，郭翔，Haas R.，Ji P.，Krongchon K.，李勇，等. Scicode: 由科学家策划的研究编码基准. 神经信息处理系统进展, 37:30624–30650, 2024.[37] H. Touvron，T. Lavril，Izacard G.，Martinet X.，Lachaux M.-A.，Lacroix T.，Rozière B.，Goyal N.，Hambro E.，Azhar F.，等. Llama: 开放且高效的基础语言模型. *arXiv*预印本*arXiv:2302.13971*, 2023.[38] K. Vodrahalli，S. Ontanon，Tripuraneni N.，徐凯，Jain S.，Shivanna R.，Hui J.，Dikkala N.，Kazemi M.，Fatemi B.，R. Anil，E. Dyer，S. Shakeri R.，Vij R.，Mehta H.，Ramasesh V.，Le Q.，Chi E.，卢勇，Firat O.，Lazaridou A.，Lespiau J.-B.，Attaluri N.，和K. Olszewska. Michelangelo: 通过潜在结构查询超越Haystacks的长期上下文评估，2024.[39] 万飞，沈伟，廖思，石勇，李超，杨哲，张俊，F. 黄，周杰，和严明. Qwenlong-l1: 通过强化学习走向长上下文大推理模型. *arXiv*预印本*arXiv:2505.17667*, 2025.[40] 王磊，高航，赵超，孙翔，戴东. 专家混合的无辅助损失负载均衡策略 . *arXiv* 预印本 *arXiv:2408.15664*, 2024.[41] 王思，于磊，高超，郑超，刘思，陆睿，唐凯，陈翔，杨杰，张哲，等. 超越80/20法则：高熵少数标记驱动有效的强化学习以实现大语言模型推理. *arXiv*预印本*arXiv:2506.01939*, 2025.[42] 王翔，李博，宋勇，徐飞飞，唐翔，祝茂哲，潘俊，宋勇，李博，J. Singh，H. H. Tran，F. 李，马睿，Zheng M.，钱博乾，张勇，赵娜，Muennighoff N.，张勇，崔博，Hui B.，J. Lin，R. Brennan，彭海，J. Ji，和G. Neubig. Openhands: 为AI软件开发者作为通才代理提供一个开放平台. 在 第十三国际学习表示会议, 2025.[43] 王勇，马翔，张刚，倪勇，Chandra A.，郭思，任伟，Arulraj A.，He X.，蒋哲，等. Mmlu-pro: 一个更鲁棒和更具挑战性的多任务语言理解基准.神经信息处理系统进展, 37:95266–95290, 2024.[44] 魏杰，Karina N.，Chung H. W.，Jiao Y. J.，Papay S.，Glaese A.，Schulman J.，和W. Fedus. 大型语言模型中的短形式事实性测量，2024.[45] 魏杰，孙卓，Papay S.，McKinney S.，韩杰，Fulford I.，Chung H. W.，Passos A. T.，Fedus W.，和A. Glaese. Browsecomp: 一个简单但具有挑战性的浏览代理基准. *arXiv*预印本*arXiv:2504.12516*, 2025.[46] 许哲，丁勇，陈伟，洪博，郭浩，王杰，杨东，廖超，郭翔，何伟，等. Agentgym: 在不同环境中进化基于大语言模型的代理. *arXiv*预印本*arXiv:2406.04151*, 2024.[47] 杨昂，李昂，杨博，张博，崔博，张宇，C. 高，黄超，吕超，等. Qwen3技术报告. *arXiv*预印本*arXiv:2505.09388*, 2025.[48] 姚思，Shinn N.，Razavi P.，和K. Narasimhan. tau-bench: 真实领域工具代理用户交互的基准. *arXiv*预印本*arXiv:2406.12045*, 2024.[49] 余启，张哲，朱睿，袁勇，左翔，岳勇，戴伟，范涛，刘刚，刘丽，等. Dapo: 一个可扩展的开源llm强化学习系统. *arXiv*预印本*arXiv:2503.14476*, 2025.[50] 曾昂，刘翔，杜哲，王哲，来海，丁明，杨哲，徐勇，郑伟，夏翔，等. Glm-130b: 一个开放的双语预训练模型. 在第十一国际学习表示会议.

[51] 张Z., 雷L., 吴L., 孙R., 黄Y., 龙C., 刘X., 雷X., 唐J., 和黄M. Safetybench: 使用多项选择题评估大语言模型的安全性。*arXiv*预印本*arXiv:2309.07045*, 2023.[52] 周J., 陆T., 米什拉S., 布拉马S., 巴苏S., 阮Y., 周D., 和侯L. 大语言模型的指令遵循评估。*arXiv*预印本*arXiv:2311.07911*, 2023.