强化学习规模化的艺术：面向大语言模型的计算

摘要与标题

原文翻译

强化学习规模化的艺术：面向大语言模型的计算

Devvrit Khatri[2]..[1], Lovish Madaan [1].[3]., Rishabh Tiwari†, Rachit Bansal†, Sai Surya Duvvuri†, Manzil Zaheer[1]†, Inderjit S. Dhillon[2], David Brandfonbrener[1], Rishabh Agarwal[1]
[1]Meta, [2]UT Austin, [3]UCL, [4]UC Berkeley, [5]Harvard University, [6]Periodic Labs *同等贡献，工作于 Meta 完成

强化学习（RL）已成为训练大语言模型（LLMs）的核心，然而该领域缺乏与预训练中已建立的可预测规模化方法论相媲美的方法。尽管计算预算迅速增长，但对于如何评估算法改进以扩展强化学习计算，尚无原则性的理解。我们呈现了首个大规模系统性研究，耗时超过 400,000 GPU小时，定义了一个用于分析和预测大语言模型中强化学习规模化的原则性框架。我们为强化学习训练拟合了S型（sigmoidal）计算-性能曲线，并对一系列常见的设计选择进行了消融研究，以分析它们对渐近性能和计算效率的影响。我们观察到：（1）并非所有配方都能产生相似的渐近性能；（2）损失聚合、归一化、课程学习和离策略算法等细节主要调节计算效率，而不会实质性地改变渐近性能；（3）稳定、可扩展的配方遵循可预测的规模化轨迹，从而能够从小规模运行中进行外推。结合这些见解，我们提出了一种最佳实践配方，名为ScaleRL，并通过在一次扩展至 100,000 GPU小时的单一强化学习运行中成功扩展并预测验证性能，证明了其有效性。我们的工作既为分析强化学习的规模化提供了一个科学框架，也提供了一个实用的配方，使强化学习训练更接近于预训练中长期以来所实现的可预测性。

深度解读

首先，让我们来剖析这个标题：“强化学习规模化的艺术：面向大语言模型的计算”。这里的关键词是“艺术”（Art）。在科学和工程领域，当一个过程被称为“艺术”时，通常意味着它依赖于直觉、经验和大量的试错，缺乏一套清晰、可重复、可预测的科学规律。作者们开宗明义地指出，当前如何通过增加计算资源（GPU算力）来提升大模型的强化学习效果，很大程度上还是一门“玄学”或“手艺活”，而非一门严谨的“科学”。这篇论文的核心目标，就是要把这门“艺术”转变为一门“科学”。

接下来看摘要，这是整篇论文的“电梯演讲”，用最短的篇幅告诉你他们做了什么、发现了什么以及贡献了什么。

1. **问题陈述**：强化学习（Reinforcement Learning, RL）对于训练顶尖大语言模型（LLMs）至关重要，但我们不知道如何科学地“堆算力”。在模型的“预训练”阶段（可以理解为模型学习海量知识的通识教育阶段），科学家们已经发现了一些“规模法则”（Scaling Laws），可以相当准确地预测投入多少算力，模型性能会提升多少。但在强化学习这个“专业技能训练”阶段，这种法则却付之阙如。这导致了巨大的资源浪费，因为公司可能会投入数百万美元的计算资源，却发现训练效果不佳，甚至崩溃。

2. **解决方案**：为了解决这个问题，研究团队进行了一项史无前例的大规模实验（耗费超过40万GPU小时，这是一个惊人的数字），并提出了一个核心工具：一个用于分析和预测强化学习效果的“原则性框架”。这个框架的核心是一个数学模型——**S型曲线（sigmoidal curve）**。你可以把它想象成学习新技能的曲线：一开始进步缓慢，然后进入快速提升期，最后随着技能掌握得差不多了，进步速度会再次放缓，趋近一个极限。这个框架就是用来精确描述模型在强化学习过程中的“学习曲线”的。

3. **核心发现**：

    - **天花板各不相同**：不同的训练方法（论文里称之为“配方”，recipe），其最终能达到的性能上限（渐近性能）是不同的。有些方法可能前期进步飞快，但很快就撞到“天花板”；而另一些方法可能起步较慢，但潜力更大，能达到更高的性能上限。

    - **效率与上限的分离**：很多技术细节，比如如何汇总计算损失、如何对数据进行标准化处理等，更多地是影响“学习效率”（即多快能达到天花板），而不太会改变“天花板”本身的高度。这为我们优化训练过程提供了重要指导：有些调整是为了“更快”，有些是为了“更高”。

    - **可预测性是关键**：稳定、优秀的训练方法，其“学习曲线”是平滑且可预测的。这意味着我们可以在训练的早期阶段，通过较少的计算资源，观察曲线的走势，然后用S型曲线模型去“外推”（extrapolate），预测出如果投入海量资源，模型最终能达到什么水平。这就像通过一个学生前几个月的学习成绩和进步趋势，来预测他高考大概能考多少分一样。

4. **最终成果**：基于以上发现，作者们整合出了一套“最佳实践配方”，命名为**ScaleRL**。为了证明这个配方和他们提出的预测框架的威力，他们进行了一次长达10万GPU小时的超大规模训练。结果完美验证了他们的理论：训练后期的实际表现，与早期数据预测出的曲线高度吻合。

总而言之，这篇论文最重要的贡献并非仅仅是发明了一个名为ScaleRL的新算法，而是提供了一套**科学的“测量工具”和方法论**。这个S型曲线框架，就像是为强化学习这门“艺术”引入了物理学中的定律和公式，让整个领域的研究者都能用一种更科学、更可预测的方式来评估和开发新的训练技术，从而推动AI能力的下一次飞跃。

图1：可预测地将强化学习计算扩展至100,000 GPU小时

原文翻译

图1：可预测地将强化学习计算扩展至100,000 GPU小时 (a) 我们在一个8B稠密模型上运行ScaleRL 10万GPU小时，在一个17B×16 MoE (Scout)模型上运行5万GPU小时。我们在独立同分布（iid）验证数据集上，对高达5万（8B模型）和1.6万（Scout MoE模型）GPU小时的通过率（mean@16）拟合了一条S型曲线（公式(1)），并分别外推至10万和4.5万GPU小时。我们对8B模型训练了7400步，对Scout模型训练了7100步，这比ProRL (Liu et al., 2025a) 大3.5倍。外推曲线（x标记）与扩展训练的实际点紧密贴合，展示了大规模计

算下的稳定性和预测拟合的准确性，确立了ScaleRL作为强化学习规模化的可靠候选者。(b) 在AIME-24上的下游评估显示，ScaleRL呈现出一致的规模化趋势，因此其泛化能力超越了训练数据分布。此外，扩大模型规模显著提升了下游任务和渐近强化学习性能。

深度解读

这张图是整篇论文的点睛之笔，是所有理论和实验最终的“成果展示”。它用两张子图直观地证明了作者们的核心论点：他们提出的ScaleRL配方和S型曲线预测框架是稳定、有效且可预测的。

(a) 子图：验证集上的可预测性

这张图展示了在“闭卷考试”中的表现。这里的“闭卷考试”指的是一个**独立同分布（iid）验证集**，你可以理解为从训练题库中随机抽取出来、但模型在训练时从未见过的一小部分题目。用这个验证集来评估，可以客观地衡量模型是否真正学到了知识，而不是死记硬背。

- **横轴（GPU Hours x 10^3）**：代表计算资源投入量，以“千GPU小时”为单位，采用对数坐标轴。对数坐标意味着横轴上从1到10的距离和从10到100的距离是一样的，这有助于我们清晰地观察早期和晚期训练的变化。
- **纵轴（Pass rate）**：代表模型在验证集上的“通过率”，即解决问题的正确率。这也是对数坐标，用于更好地展示性能的相对提升。
- **曲线和数据点**：
  - **实心点（training points）**：这是模型在训练过程中的实际表现数据点。
  - **实线（fitted curve）**：这是研究人员利用**训练前期**的数据点（例如，图中只用到了前5万GPU小时的数据），通过他们的S型曲线公式拟合出的一条预测曲线。
  - **x标记（extended training points）**：这是模型在**训练后期**（5万到10万GPU小时之间）的实际表现。
  - **虚线（extrapolated curve）**：这是由实线延伸出来的**预测部分**。

这张图最震撼的地方在于：代表着“未来实际表现”的“x”标记，几乎完美地落在了代表着“早期预测”的虚线上。这雄辩地证明了，研究者们提出的S型曲线框架确实有“未卜先知”的能力。他们可以在只花费一半计算资源的时候，就相当准确地预测出花费全部资源后模型能达到的最终性能。这对动辄耗资数百万美元的AI训练来说，意义非凡。

图中还比较了两种模型：一个80亿参数的稠密模型（ScaleRL-8B Dense）和一个更先进、规模更大的混合专家模型（ScaleRL-17Bx16 MoE）。可以看到，更大更强的模型，其性能曲线整体位于上方，意味着它的“天花板”更高。

(b) 子图：下游任务上的泛化能力

这张图展示了在“开卷应用题”中的表现。这里的“应用题”指的是一个名为**AIME-24**的真实世界数学竞赛数据集。这个数据集的题目风格、难度分布都可能和训练数据（Polaris-53k）有所不同，所以能测试模型的**泛化能力**（generalization），即将在一个领域学到的知识应用到另一个相关领域的能力。

> **结果**：图(b)显示，在AIME-24这个更具挑战性的任务上，性能的提升趋势与图(a)中的趋势高度一致。投入更多计算资源，模型在真实竞赛题上的表现也越好。并且，规模更大的MoE模型（绿色曲线）再次展现了碾压性的优势。

综合解读：

图1通过两个维度传达了强有力的信息：

1. **科学的可预测性**：图(a)证明了强化学习的规模化可以从“艺术”变为“科学”。我们不再需要“赌博式”地投入海量资源，而是可以通过科学的建模，在早期阶段就对最终结果做出可靠预测。

2. **真实世界的有效性**：图(b)证明了这种性能提升不是“虚假的繁荣”（即只在训练数据上表现好），而是能实实在在地转化为解决真实世界难题的能力。

3. **模型规模的重要性**：两张图都清晰地表明，在拥有了稳定可扩展的训练方法（ScaleRL）之后，提升模型自身的基础规模（从8B到17Bx16 MoE）是突破性能上限最有效的方式之一。这再次印证了AI领域“越大越好”（Bigger is Better）的普遍规律，但前提是你要有能力驾驭好这个“大家伙”，而ScaleRL正是提供了这种驾驭能力。

1. 引言

原文翻译

强化学习（RL）计算的规模化正成为推动大语言模型（LLMs）发展的关键范式。虽然预训练为模型奠定了基础，但随后的强化学习训练阶段解锁了当今许多最重要的LLM能力，从测试时思考（OpenAI, 2024; Guo et al., 2025）到智能体能力（Kimi Team et al., 2025a）。例如，Deepseek-R1-Zero 使用了 100,000 H800 GPU小时进行强化学习训练——占其预训练计算量的3.75%（Guo et al., 2025）。这种强化学习计算的急剧增加在最前沿的LLM代际中被放大，从o1到o3增长了超过10倍（OpenAI, 2025），从Grok-3到Grok-4也有类似的飞跃（xAI Team, 2025）。

虽然用于LLM的强化学习计算已经大规模扩展，但我们对如何扩展强化学习的理解却未能跟上；其方法论仍然更像是一门艺术而非科学。最近在强化学习领域的突破主要是由关于新颖算法的孤立研究（例如，Yu et al. (DAPO, 2025)）和特定模型的训练报告（例如，MiniMax et al. (2025) 和 Magistral (Rastogi et al., 2025)）所驱动。关键在于，这些研究提供了针对特定情境的临时解决方案，但并未说明如何开发能随计算资源扩展的强化学习方法。这种规模化方法论的缺乏扼杀了研究进展：由于没有可靠的方法来预先识别有前景的强化学习候选方案，进展被绑定在需要进行大规模实验上，这使得大多数学术界被边缘化。

这项工作借鉴了预训练中成熟的规模法则（scaling laws）概念，为强化学习规模化的科学奠定了基础。虽然预训练已经收敛到能够随计算资源可预测地扩展的算法配方（Kaplan et al., 2020; Hoffmann et al., 2022; Owen, 2024），但强化学习领域缺乏一个明确的标准。因此，强化学习从业者面临着一系列令人眼花缭乱的设计选择，使得如何扩展和扩展什么这些基本问题悬而未决。

深度解读

这段引言为整篇论文搭建了舞台，深刻地阐述了研究的动机和紧迫性。作者们通过三个层次的论述，清晰地描绘了当前AI领域面临的一个核心矛盾。

第一层：强化学习（RL）已成为AI能力突破的“胜负手”

作者首先指出，如果说“预训练”是给大模型灌输海量知识，构建起一个庞大的“知识图书馆”，那么“强化学习”就是教模型如何**运用**这些知识去思考、推理和解决问题。这就像一个学生，背完了所有课本（预训练），还需要通过做题、考试和老师的反馈（强化学习）来学会如何灵活运用知识。当今最尖端的能力，如“测试时思考”（模型在回答问题前会自己先进行一番草稿式的推理）和“智能体能力”（模型能像一个自主的代理人一样规划和执行复杂任务），都是通过强化学习解锁的。

为了强调其重要性，作者列举了惊人的数据：Deepseek-R1-Zero模型在强化学习上花费了10万GPU小时，占其预训练总计算量的3.75%，这是一个相当大的比例。更重要的是，从OpenAI的o1到o3模型，以及xAI的Grok-3到Grok-4模型，用于强化学习的计算资源都呈10倍以上的爆炸式增长。这清晰地表明，业界巨头们已经达成共识：**未来的竞争焦点，在于谁能更好地利用强化学习来打磨和提升模型的高级智能**。预训练的“军备竞赛”远未结束，但强化学习的“精细化调优”竞赛已经白热化。

第二层：RL的实践现状是“艺术”，而非“科学”，这造成了巨大问题

接着，作者笔锋一转，指出了一个残酷的现实：尽管我们投入的资源越来越多，但我们对如何有效利用这些资源进行强化学习的理解却非常落后。现状是“**艺术而非科学**”。这意味着成功往往依赖于少数顶尖工程师的“祖传秘方”和“炼丹手艺”，充满了不确定性。今天的成功经验，明天换个模型或任务可能就完全失效。

这种“艺术化”的现状带来了两个严重问题：

1. **研究进展缓慢且昂贵**：现有的研究成果，如DAPO、MiniMax等，都像是“孤立的岛屿”，它们提供了在特定场景下的解决方案，但没有提供一个通用的“航海图”，告诉大家如何系统性地探索和开发能够随计算资源投入而持续变强的RL方法。

2. **学术界被边缘化**：最致命的是，这种依赖大规模、高成本实验的“试错”模式，фактически将绝大多数没有雄厚财力支持的学术研究机构排除在外。如果验证一个新想法的“门票”就是数百万美元的计算资源，那么创新将只属于少数几个科技巨头。这严重阻碍了整个AI领域的健康发展和知识共享。

第三层：我们的解决方案——借鉴预训练的成功经验，建立RL的“科学”

最后，作者提出了他们的雄心壮志：为强化学习领域建立一套像预训练领域那样成熟的“**规模法则**”（Scaling Laws）。在预训练中，研究者已经能够用简洁的数学公式来描述模型大小、数据量、计算量和最终性能之间的关系，实现了高度的可预测性。作者的目标，就是为强化学习这个“混乱的西部”引入同样的“法律与秩序”。

他们指出，当前RL从业者面对无数个技术选择（比如用哪种损失函数、如何设置参数等），就像走进一个没有地图的迷宫。这篇论文就是要绘制这张地图，回答“**如何扩展**”和“**扩展什么**”这两个最根本的问题。他们即将提出的S型曲线框架，就是这张地图的核心，是他们将RL从“艺术”推向“科学”的宣言。

2. 预测性框架

原文翻译

为了解决这些问题，我们使用一个类S型（sigmoid-like）的饱和曲线来建立强化学习性能的预测框架，该曲线描述了在独立同分布（iid）验证集上的预期奖励（RC）与训练计算量（C）之间的关系：

RC−R0=1+(Cmid/C)B(A−R0)

奖励增益=渐近奖励增益×1+(Cmid/C)B1(计算效率)

（固定模型和训练数据） (1)

其中 0<A<1 代表渐近通过率，B>0 是一个决定计算效率的缩放指数，Cmid 设定了强化学习性能曲线的中点。这些参数的示意图解释在图3中提供。

这个在公式(1)中的框架允许研究人员从较低计算量的运行中外推性能至较高的计算预算，使他们能够评估强化学习方法的可扩展性，而无需承担将每个实验运行到其计算极限的成本。

在这一框架的指导下，我们开发了ScaleRL，一个能随计算资源可预测地扩展的强化学习配方。在一次耗时100,000 GPU小时的大规模训练中，我们展示了ScaleRL的性能与我们框架预测的规模曲线紧密匹配（图1）。关键的是，仅从训练初始阶段外推的规模曲线与最终观察到的性能非常吻合，证实了我们的框架在极端计算规模下的预测能力。

ScaleRL的设计基于一项全面的强化学习规模化实证研究，该研究耗时超过400,000 GPU小时（在Nvidia GB200 GPU上）。这项研究在8B模型参数规模上探索了众多设计选择，其中单次运行使用高达16,000 GPU小时，比我们最大规模的训练实验便宜6倍。这项调查得出了三个关键原则：

- **强化学习性能天花板并非普遍一致**：当我们为不同方法扩展训练计算量时，它们会遇到不同的可实现性能（A）的上限。这个上限可以通过损失类型和批量大小等选择来改变。
- **拥抱“惨痛的教训”**：在小计算预算下看起来更优的方法，在外推到大计算规模时可能会变得更差（图2）。我们仍然可以通过使用我们的框架（公式(1)）从早期训练动态中估计规模参数（A, B）来识别可扩展的方法。

深度解读

这是整篇论文的理论核心，作者在这里正式亮出了他们最关键的工具——**S型饱和曲线公式**。这个公式看似复杂，但如果你理解了它的物理意义，就会发现它非常直观，并且威力巨大。让我们用一个生动的比喻来彻底拆解它。

把模型训练想象成一个学生备考

- **C (Compute)**：代表“计算量”，也就是学生**投入的学习总时长**（比如，总共复习了多少小时）。
- **RC (Reward at Compute C)**：代表学生在学习了 C 小时后，参加模拟考的**预期分数**。
- **R0 (Initial Reward)**：代表学生在开始系统复习前的**初始分数**（摸底考试成绩）。
- **RC−R0**：这就是学生通过学习获得的**分数提升**。

现在，我们来看公式的核心部分，它描述了分数提升是如何发生的：

分数提升=(最大可能的分数提升)×(学习效率因子)

1. 渐近奖励增益 (Asymptotic Reward Gain): A−R0

- **A (Asymptotic Performance)**：这是这篇论文最重要的概念之一，代表“**渐近性能**”，也就是**理论上的最高分**。无论这个学生多么努力，复习多么久，他的分数最终会无限接近但不会超过这个值 A。这代表了这个学生（或这个模型+训练方法组合）的**潜力上限**或**性能天花板**。例如，对于某个学生，他的潜力天花板可能是95分。
- **A−R0**：这就是这位学生从开始复习到发挥出全部潜力，所能获得的**最大总提升分数**。如果他摸底考50分，潜力天花板是95分，那么这个值就是45分。

2. 计算效率 (Compute Efficiency) 因子

这个因子是一个分式，1+(Cmid/C)B1，它决定了学生达到他潜力天花板的**速度**。

- **Cmid (Midpoint Compute)**：代表“**学习中点**”。这是指学生达到他**最大总提升分数一半**时，所需要的学习时长。比如，总共能提升45分，那么达到提升22.5分（考到72.5分）时花了多少小时，这个时长就是 Cmid。Cmid 的值越小，说明学生“开窍”越早，能越快地进入分数的快速增长期。
- **B (Scaling Exponent)**：代表“**学习效率指数**”。这个值越大，说明学生的学习方法越高效。在S型曲线上，一个大的 B 值会使曲线中间的上升阶段**非常陡峭**。这意味着一旦“开窍”（越过 Cmid 附近），他的分数会像坐火箭一样飞速提升。

这个框架的革命性意义

这个公式框架的真正威力在于，它将一个模糊的“性能提升”问题，分解成了三个可以被精确测量的、具有明确物理意义的参数：

- **A (潜力)**：决定了我们**最终能走多远**。
- **B (效率)**：决定了我们**走得有多快**（特指快速上升期的速度）。
- **Cmid (起点)**：决定了我们**多早开始快速前进**。

通过这个框架，研究人员可以提出并回答比“方法X比方法Y好吗？”更深刻的问题。他们可以问：

- “方法X是**提高了模型的潜力天花板（A）**，还是仅仅**让模型学得更快（改变了B或Cmid）**？”
- “为什么有些方法在早期看起来很厉害，但后期却停滞不前？”答案可能就是，它的 A 值很低，潜力有限。这就是作者提到的“**惨痛的教训**”（The Bitter Lesson）：一个早期表现优异但 A 值低的方法，远不如一个早期表现平平但 A 值高的方法有前途。

最终，这个框架使得**预测**成为可能。研究者不再需要完整地跑完一个耗时10万GPU小时的实验。他们可以先跑1-2万小时，获得足够的数据点来拟合这条S型曲线，估算出 A, B, Cmid 的值。然后，他们就可以用这个公式来预测，如果继续投入8万小时，最终的性能会达到哪里。这正是图1所展示的惊人能力，也是将强化学习规模化从“艺术”变为“科学”的关键一步。

图2：ScaleRL比主流的RL方法更具可扩展性

原文翻译

图2：ScaleRL比主流的RL方法更具可扩展性。 我们在独立同分布（iid）验证数据集上，对一些常用的训练配方，如DeepSeek (GRPO) (Guo et al., 2025)、Qwen-2.5 (DAPO) (Yu et al., 2025)、Magistral (Rastogi et al., 2025) 和 Minimax-M1 (MiniMax et al., 2025)，拟合了S型曲线（公式1），并与ScaleRL进行比较。ScaleRL超越了所有其他方法，实现了 A=0.61 的渐近奖励。星号表示评估点；实线曲线显示了用于拟合的范围内的拟合曲线；虚线曲线则外推至该范围之外。我们通过将每种方法运行更长时间（“x”标记）来验证其可预测性，对于像ScaleRL和MiniMax这样的稳定配方，这些标记与外推曲线紧密对齐。所比较的各个配方的进一步描述在附录A.16中给出。

深度解读

这张图是论文中一个极具说服力的“擂台赛”展示。作者们将他们精心打造的ScaleRL配方，与当前业界几种主流或知名的强化学习方法进行了正面比较。这张图不仅证明了ScaleRL的优越性，更重要的是，它生动地诠释了前文提到的“惨痛的教训”（The Bitter Lesson）。

图表元素解析

- **横轴与纵轴**：与图1相同，分别是代表计算资源投入的“GPU小时”（对数坐标）和代表模型性能的“通过率”（对数坐标）。
- **不同颜色的曲线**：每种颜色代表一种不同的强化学习“配方”或方法。

- **图例中的A和B值**：图例中为每种方法标注了拟合出的S型曲线的两个关键参数：A（渐近性能，即潜力天花板）和 B（学习效率指数）。

- **星号（*）与叉号（x）**：星号是用于拟合曲线的实际数据点，叉号是后期扩展训练的实际数据点，用来验证预测的准确性。

核心观察与洞见

1. **ScaleRL的全面胜利**：从图上可以清晰地看到，代表ScaleRL的蓝色曲线在所有比较方法中，最终达到了最高的性能水平。其图例显示 A=0.610，与MiniMax持平，但显著高于Magistral (A=0.535)、Qwen2.5 (A=0.515) 和 DeepSeek (A=0.490)。这表明ScaleRL拥有最高的“潜力天花板”。同时，它的 B 值为1.97，也是所有方法中最高的之一，意味着其学习效率也非常高。

2. **“惨痛的教训”的生动案例**：请特别关注**DeepSeek (GRPO)** 的紫色曲线。在训练的极早期（大约2000 GPU小时之前），它的性能是所有方法中最高的，看起来非常有前途。然而，它的上升势头很快就减弱了，并且早早地趋于平缓，最终饱和在一个非常低的性能水平上（A=0.490）。这完美地诠释了“惨痛的教训”：一个在小规模、短时间实验中表现最好的方法，在大规模、长时间的尺度下，可能恰恰是潜力最差的。如果不使用S型曲线框架进行长远预测，研究者很可能会被早期结果误导，从而选择一个错误的、没有前途的技术路线。

3. **稳定性的重要性**：注意看，对于ScaleRL（蓝色）和MiniMax（橙色），代表后期实际性能的“x”标记都很好地落在了虚线预测曲线上，这说明这两种方法是**稳定且可预测的**。相比之下，其他一些方法（图中未明确标出，但作者在文中暗示）可能在扩展训练后表现出不稳定性，导致实际性能偏离预测。这说明，一个好的RL配方不仅要性能高，还必须足够稳定，才能保证在投入海量资源后，训练过程不会“翻车”。

4. **区分“潜力”与“效率”**：比较**ScaleRL**（蓝色）和**Magistral**（绿色）。ScaleRL的 A 值（0.610）远高于Magistral（0.535），但Magistral的 B 值（1.44）也不低。这说明Magistral也是一个不错的配方，但它的根本问题在于“潜力上限”不够高。通过S型曲线框架，我们可以清晰地诊断出不同方法的优劣所在，是潜力不足（A低），还是效率不高（B低）。

总结

图2不仅是一张性能对比图，更是一次方法论的胜利。它展示了S型曲线框架作为一个强大的“诊断工具”，能够帮助我们超越表面的、短期的性能数据，洞察每种方法内在的、长期的**可扩展性**（scalability）。它告诉所有AI研究者一个道理：在评估一个新算法时，不要只看它在小规模实验上跑得多快多好，而要看它的“潜力天花板”A有多高，以及它的规模化轨迹是否稳定可预测。ScaleRL正是在这个更科学、更具前瞻性的评估体系下脱颖而出的胜利者。

原文翻译

重新评估普遍认知：普遍认为可以提升峰值性能的干预措施（例如，损失聚合、数据课程、长度惩罚、优势归一化）主要调整了计算效率（B），而没有显著改变性能天花板。

基于这些见解，ScaleRL通过整合现有方法，而非发明新方法，实现了可预测的规模化。具体来说，ScaleRL结合了异步Pipeline-RL设置（§3.1）、强制长度中断、截断重要性采样RL损失（CISPO）、提示级损失平均、批次级优势归一化、logits处的FP32精度、零方差过滤和无正样本重采样，每个组件的贡献都在“留一法”消融实验中得到验证，每次运行消耗16,000 GPU小时。

ScaleRL不仅可预测地扩展，还建立了新的技术水平（图2）——与已建立的RL配方相比，它实现了更高的渐近性能和计算效率。此外，当在多个训练轴上增加计算量时（§5），包括2.5倍大的批量大小、长达32,768个token的更长生成长度、使用数学和代码的多任务RL，以及更大的MoE模型（Llama-4 17B×16），ScaleRL都保持了可预测的规模化；其优势能够持续迁移到下游任务。总的来说，这项工作为经济高效地预测新RL算法的可扩展性建立了一套严谨的方法论。

2. 预备知识与设置

我们考虑使用LLMs进行强化学习，其中提示从数据分布D中采样。我们的设置遵循跨GPU的生成器-训练器分离：一部分GPU（生成器）使用优化的推理核心进行高吞吐量的轨迹生成，而其余的GPU（训练器）运行训练后端（FSDP）并更新参数。我们用$\pi_{gen}^{\theta_{old}}表示在生成器后端上的旧策略（带有参数\theta_{old}），用\pi_{train}^{\theta}表示在训练器后端上的模型。对于每个提示，生成器GPU上的旧策略\pi_{gen}^{\theta_{old}}产生候选的补全，然后为这些补全分配标量奖励。策略优化通过最大化一个裁剪的替代目标来进行，该目标在x \sim D和从\pi_{gen}^{\theta_{old}}$生成的轨迹上取期望。

训练方案 所有实验都在推理领域的RL上进行，模型会生成一个用特殊标记（`<think>` `</think>`）包围的思考轨迹和一个最终解决方案。除非另有说明，训练使用的序列长度为16,384个token：12,288个用于思考，2,048个用于解决方案，另外2,048个用于输入提示。我们采用12,288的思考预算以加快迭代速度，并在第5节中表明，当使用更大的思考预算（32,768）进行训练时，ScaleRL的外推预测仍然保持准确。对于数学RL实验，我们使用Polaris-53K数据集（An et al., 2025），批量大小为768（48个提示，每个提示16个生成）。在我们的设置中，扩展RL计算对应于在训练提示上运行多个轮次（epochs）。更多关于训练的细节，包括SFT和超参数，请参见附录A.3。

深度解读

这段内容是论文的技术“地基”，它详细说明了作者们进行所有实验所采用的基础架构、流程和设定。对于高中生读者来说，理解这些细节有助于你了解一个顶级的AI研究实验是如何被设计和执行的。我们可以把它分解成几个关键部分来理解。

1. 核心架构：生成器-训练器分离 (Generator-Trainer Split)

这是一种在大规模AI训练中非常常见且高效的架构。你可以把它想象成一个**“学生团队”和“教师团队”**的协同工作模式：

- **生成器 (Generators)**：这是一大群专门负责“做题”的GPU。它们就像是学生，拿到一个问题（prompt），使用当前版本的“解题方法”（旧策略 $\pi_{gen\theta old}$）来生成大量的解题过程和答案（rollouts/completions）。为了让它们“做题”速度飞快，这些GPU上运行的是高度优化的推理代码。

- **训练器 (Trainers)**：这是另一小组负责“批改作业和教学”的GPU。它们接收来自“学生团队”的大量答案，根据一个评分标准（奖励模型）给每个答案打分，然后分析哪些解法好、哪些不好。基于这些分析，它们会更新“教学大纲”（模型参数 θ），形成一个新版本的“解题方法”（新策略 $\pi_{train\theta}$）。

- **流程**：训练器将更新后的“教学大纲”广播给所有生成器，生成器们就用这个新方法继续“做题”。如此循环往复。

这种分离架构的好处是**专业化和高效率**。生成器可以专注于快速产出数据，训练器可以专注于复杂的梯度计算和参数更新，两者并行工作，极大地提升了整个训练系统的吞吐量。

2. 任务设定：带思考过程的推理 (Reasoning with Thinking Trace)

作者们选择的任务领域是“推理”，特别是需要模型展现其思考过程的任务。

- `<think>... </think>` **结构**：模型被要求在给出最终答案前，先生成一段被特殊标记包围的“思考轨迹”（Chain-of-Thought, CoT）。这就像在数学考试中，老师不仅看你的最终答案，更要看你的解题步骤。这使得研究者可以更好地理解模型的“思维过程”，并且奖励模型也可以基于解题步骤的正确性来给出更精确的反馈。

- **序列长度 (Sequence Length)**：总长度为16,384个token（可以粗略理解为一个token约等于一个单词或汉字），这个长度被精确地划分：2048给问题本身，12288给模型的“草稿纸”（思考过程），2048给最终的解答。这是一个非常长的上下文窗口，允许模型进行非常复杂的、多步骤的推理。作者提到，他们先用一个较短的思考预算（12,288）来快速进行各种实验，后续会证明他们的结论在更长的预算（32,768）下依然成立。

3. 训练数据和规模

- **数据集**：主要使用一个名为Polaris-53K的数学问题数据集。这是一个专门用于训练模型推理能力的高质量数据集。

- **批量大小 (Batch Size)**：每次参数更新，训练器会处理一个包含768个答案的“大批量作业”。这个大批量由48个不同的问题组成，每个问题都让模型生成了16个不同的答案。让模型对同一个问题生成多个答案，可以增加探索的多样性，让模型看到更多种可能的解题路径。

- **扩展计算的含义**：在这项研究中，“增加计算量”意味着让模型在整个训练数据集上**反复学习更多轮次（epochs）**。就像一个学生为了备考，把整本习题集从头到尾做了好几遍一样。

通过这些详细的设置，作者们确保了他们的实验是在一个高度规范、可控且贴近前沿应用场景的环境下进行的。这为他们后续所有发现的可靠性和说服力奠定了坚实的基础。

原文翻译

基础RL算法 我们在§3中的出发点是一个“基础”算法，它类似于没有KL正则化项的GRPO (Shao et al., 2024)，这与大规模训练报告（Rastogi et al., 2025; MiniMax et al., 2025）的做法一致。此外，我们加入了非对称的DAPO裁剪（Yu et al., 2025），因为它被广泛用作避免熵崩溃和维持输出多样性的默认方法。

对于给定的提示x，旧策略$\pi_{gen}(\theta_{old})生成G个候选补全{y_i}_{i=1}^G$，每个都被赋予一个标量奖励ri。我们计算优势Ai和组归一化的优势，使用：

A^i=ri−mean({rj}j=1G)

A^iG=A^i/(std({rj}j=1G)+ϵ)

每个长度为$|y_i|的补全y_i在token级别贡献重要性采样（IS）比率\rho_{i,t}(\theta)$，并带有非对称的上下裁剪阈值，类似于DAPO（Yu et al., 2025）：

ρi,t(θ):=πgenθold(yi,t∣x,yi,<t)πtrainθ(yi,t∣x,yi,<t)=πgenθold(yi,t)πtrainθ(yi,t)

clipasym(ρ,ϵ−,ϵ+):=clip(ρ,1−ϵ−,1+ϵ+)

(2)

我们在样本级别上聚合损失，即在对样本进行平均之前，先对每个样本的token损失进行平均。替代目标由下式给出：

J(θ)=Ex∼D,{yi}i=1G∼πgenθold(·∣x)G1i=1∑G∣yi∣1t=1∑∣yi∣min(ρi,t(θ)A^iG,clipasym(ρi,t(θ),ϵ−,ϵ+)A^iG)
(3)

控制生成长度 为了防止推理输出的长度在训练过程中爆炸，这会损害训练的稳定性和效率，我们使用“中断”（interruptions）（GLM-V Team et al., 2025; Yang et al., 2025）。这种方法通过附加一个结束思考的短语（例如，“”）来强制停止过长的生成，向LLM发出信号，让其终止推理并产生最终答案。我们稍后在第4节中会重新审视这个选择，并将其与惩罚长生成的长度惩罚（length-penalty）方法（Yn et al., 2025; Kimi Team et al., 2025b）进行比较。

深度解读

这一部分定义了作者们进行系统性比较的“**参照物**”或“**起点**”——一个基础版的强化学习算法。选择一个好的起点至关重要，它必须足够有代表性，能反映当前领域内比较通用的做法，这样基于它进行的改进和比较才有意义。

1. 基础算法的选择：融合主流思想

作者们没有从零开始发明一个算法，而是聪明地选择了一个融合了当时几种主流思想的“混合体”作为他们的“基础版”（base algorithm）。

- **骨架是GRPO**：GRPO是一种在LLM推理任务中很流行的RL算法。它的核心思想是，对于同一个问题产生的多个答案，不去看它们的绝对分数，而是看它们在这个“小组”内的**相对好坏**。

- **移除了KL正则化**：这是一个技术细节，但反映了作者们紧跟前沿实践。KL正则化是传统RL中用来防止新策略偏离旧策略太远的一种技术，但在最新的大规模训练中，很多团队发现去掉它效果可能更好。

- **加入了DAPO的非对称裁剪**：DAPO是GRPO的一个重要改进。它引入了一种更聪明的“裁剪”（clipping）机制。这个机制的作用是限制单次更新的步子不要迈得太大，防止模型“学跑偏”。DAPO的非对称性意味着，对于好的行为（正奖励），鼓励的幅度可以大一些；对于坏的行为（负奖励），惩罚的幅度则要小心控制。这被认为有助于**维持模型输出的多样性**，防止模型思维僵化，只会用一种方式解决问题（即熵崩溃）。

2. 核心数学概念的通俗解释

这里的公式是强化学习的核心，让我们用之前“老师教学生”的比喻来理解它们。

- **优势 (Advantage) A^i**：假设老师让学生用16种方法解同一道题。有些方法解对了（奖励 ri=1），有些解错了（奖励 rj=0）。**优势**就是衡量某一种解法**比这次所有解法的平均水平好多少**。如果16种方法里有8种对了，平均分是0.5。那么一个正确的解法，其优势就是 1−0.5=0.5；一个错误的解法，优势就是 0−0.5=−0.5。这告诉模型，应该学习那些“超出平均水平”的行为。

- **组归一化优势 (Group-normalized Advantage) A^iG**：这个更进了一步，它不仅考虑了平均水平，还考虑了这次所有解法分数的“波动范围”（标准差 std）。如果所有解法得分都差不多（比如一半对一半错），那么标准差就大，归一化后的优势值就小。这意味着在“情况复杂、好坏难分”的时候，模型的学习信号会弱一些，表现得更“谨慎”。这是一种稳定训练的技巧。

- **重要性采样比率 (Importance Sampling Ratio) ρi,t(θ)**：这是RL中的一个关键概念。还记得我们有“做题”的学生（生成器，用旧策略 πgenθold）和“教学”的老师（训练器，用新策略 πtrainθ）吗？学生是用旧方法做的作业，但老师要用这些作业来指导**新方法**的学习。ρ 就是一个“**修正系数**”，它衡量对于某个具体的解题步骤（token yi,t），新旧两种方法想到一块儿去的概率比。如果比值接近1，说明新旧方法思路一致；如果远大于1，说明新方法更倾向于这么做。

- **替代目标 (Surrogate Objective) J(θ)**：这就是模型要优化的最终“学习目标函数”。可以理解为，模型的目标是**最大化所有“好的”解题步骤被采纳的概率，同时最小化所有“坏的”解题步骤被采纳的概率**。这个目标函数里有一个`min`函数和`clip`函数，它们的作用就是之前提到的“裁剪”，确保模型每次学习的步子不会太大，防止“矫枉过正”，从而稳定学习过程。

3. 控制生成长度：一个棘手的工程问题

最后，作者提到了一个非常实际的问题：在RL训练中，模型有时会“走火入魔”，为了解决一个问题生成越来越长的思考过程，甚至无限循环下去。这不仅浪费计算资源，还会导致训练不稳定。作者们的基础版算法采用了一种简单粗暴但有效的方法——“**强制中断**”。就像考试时间到了，监考老师会说“停笔！”，强制你结束思考，开始写最终答案。这确保了训练过程的效率和可控性。作者也预告了，这只是其中一种方法，他们后续会把它和另一种更“温和”的方法——“长度惩罚”（即对太长的答案进行扣分）进行比较。

2.1 预测性计算规模化与曲线拟合

原文翻译

与通常使用幂律来拟合预测曲线的预训练不同，我们用一个S型函数（公式(1)）来建模通过率与log(计算量)的关系。我们这样做是因为我们凭经验发现S型拟合比幂律要鲁棒和稳定得多，我们在附录A.4中对此进行了进一步讨论。此外，我们的选择与先前使用类S型幂律来捕捉有界指标（如准确率）的工作是一致的（Ruan et al., 2024; Srivastava et al., 2022）。

与预训练研究（Li et al., 2025b; Porian et al., 2025）类似，我们发现排除极早期的低计算量阶段可以产生更稳定的拟合，在此之后，训练会遵循一个可预测的轨迹。除非另有说明，我们所有的规模化拟合都从约1.5k GPU小时后开始。拟合过程的更多细节在附录A.5中提供，我们的曲线拟合的鲁棒性在附录A.7中讨论。

解读规模曲线 直观地说，S型曲线捕捉了饱和回报的规律：在低计算量区域增长缓慢，在中等范围的有效规模化区域急剧加速，然后在高计算量区域饱和。我们还在图3中提供了对S型曲线参数A、B和$C_{mid}的示意图解释。我们看到B和C_{mid}$主要影响运行的效率，而A表示在大计算规模下达到的渐近性能。这些参数的进一步讨论在附录A.8中提供。

在留出验证集上的规模曲线 与预训练实践（Hoffmann et al., 2022; Porian et al., 2025）一致，我们在分布内的验证数据上测量预测性能。由于我们的训练运行跨越多个轮次，我们从Polaris-53k数据集中随机选出1,000个提示作为验证集，并使用其余的进行训练。规模曲线是在验证点上拟合的，这些验证点测量的是每100个训练步骤，在1,000个留出提示上每个提示16次生成的平均通过率。

深度解读

这一节详细阐述了作者们进行预测性建模的“方法论”，解释了他们为什么选择S型曲线，以及如何具体操作。

1. 为什么选择S型曲线，而不是预训练中常用的幂律（Power Law）？

这是一个非常关键的方法论抉择。在LLM的预训练领域，一个著名的“规模法则”是幂律，它大致表明，模型的损失会随着计算量、模型大小或数据量的指数级增加而线性下降（在对数-对数坐标系中）。简单来说，就是“大力出奇迹”，投入越多，效果越好，且这种关系在很大范围内是可预测的。

但作者们指出，在强化学习这个场景下，幂律模型并不适用，S型曲线是更好的选择。原因有三：

- **指标有界性**：RL的性能指标通常是“通过率”或“准确率”，这是一个有上限的指标（最高100%）。幂律模型在理论上可以无限增长，不适合描述这种会“饱和”的现象。而S型曲线天生就是用来描述这种从0增长到某个上限的过程的，比如人口增长、化学反应速率等，非常契合。

- **经验上的鲁棒性和稳定性**：作者们通过实验发现（详见附录A.4），用幂律去拟合RL的性能曲线，结果非常不稳定，对拟合数据的范围选择极为敏感，预测结果可能谬以千里。而S型曲线的拟合结果则稳定得多，预测也更准确。

- **数据利用率**：幂律通常只在计算量达到一定规模后才成立，这意味着拟合时需要丢弃大量的早期训练数据。而RL的训练总步数远少于预训练，数据点本来就很宝贵，再丢弃一部分就更难进行准确拟合了。S型曲线则能更好地利用从早期到晚期的所有数据。

2. 如何进行拟合：排除早期噪声，关注可预测轨迹

作者们借鉴了预训练研究的一个经验：在训练的最最开始阶段，模型的性能变化可能非常剧烈且不规律，就像一个刚学开车的新手，起步时可能会熄火、急刹，动作很不稳定。因此，为了得到更可靠的长期趋势预测，他们会**忽略掉最初约1500 GPU小时的数据**，从训练进入一个相对平稳的“可预测轨迹”后才开始拟合曲线。这是一种科学研究中常见的、为了抓住主要矛盾而忽略次要噪声的合理做法。

3. 在哪里测量性能：留出验证集的重要性

这是一个关于科学实验严谨性的关键点。模型的性能不能在它天天练习的“训练集”上测量，因为那可能会导致“应试教育”——模型只是死记硬背了训练题的答案，而没有真正学会解题方法。

因此，必须在一个模型从未见过的“**留出验证集**”（held-out validation set）上进行测量。这个验证集和训练集来源相同（“同分布”），就像是从同一本习题集里抽出的、但老师没讲过的题目。在这个集合上的表现，才能真实反映模型的**泛化能力**。

作者们严格遵循了这一原则：他们从53000个问题的总数据集中，随机预留了1000个问题作为“考场”，专门用来测试和拟合性能曲线。模型在训练过程中，每隔100步就会被拉到这个“考场”进行一次模拟考试（对每个问题生成16个答案，计算平均通过率），记录下的“考分”就是用来拟合S型曲线的数据点。这确保了他们评估的是模型的真实能力，而非记忆力。

图3：解读公式(1)

原文翻译

图3：解读公式(1)。 我们提供一个示例拟合，说明参数A、B和C_{mid}的作用。C_{mid}决定了达到总增益一半时的计算点——值越小对应着向渐近线攀升得越快。B控制曲线的陡峭程度，值越大表示效率越高。A代表在大计算规模下达到的渐近性能。进一步的讨论在附录A.8中提供。

深度解读

这张图是S型曲线公式(1)的“可视化说明书”，它非常直观地解释了A, B, C_{mid}这三个核心参数各自扮演的角色。对于理解这篇论文的方法论至关重要。让我们再次借用“学生备考”的比喻来深入解读这张图。

- **横轴 (Compute C)**：学生投入的学习总时长（对数坐标）。
- **纵轴 (Expected Reward RC)**：学生在学习了 C 小时后的预期分数。

参数解析：

1. **A (Asymptotic reward)：渐近性能 / 潜力天花板**
    - **图中位置**：曲线在图的右侧无限接近的那条水平虚线。
    - **物理意义**：这代表了该学生（或模型+训练方法的组合）所能达到的**理论最高分**。无论再投入多少学习时间，分数也只能无限逼近这个值。参数 A 是决定一个方法**好坏的最根本指标**。一个 A 值为0.9的方法，本质上就比一个 A 值为0.7的方法更有潜力。在RL算法竞赛中，选择一个能**提升 A 值**的改进，远比那些只能加快学习速度的改进更有价值。
2. **Cmid (compute at 50% of total gain)：学习中点**
    - **图中位置**：曲线上升到“一半高度”时，所对应的横坐标位置。这里的“一半高度”不是指分数的一半，而是指**分数提升量**的一半，即达到 (A+R0)/2 的位置。
    - **物理意义**：这代表了学生**进入“开窍”状态所需的时间**。Cmid 的值越小（越靠左），说明学生越早进入分数的快速增长期。在比较两种潜力 (A) 差不多的学习方法时，Cmid 更小的那一个通常更受欢迎，因为它能“更快见效”，在有限的备考时间内达到更高的分数。它衡量的是**起步效率**。
3. **B (Scaling exponent)：学习效率指数**
    - **图中位置**：这个参数不直接体现在某个点上，而是体现在曲线**中间部分的陡峭程度**上。
    - **物理意义**：B 值越大，曲线在 Cmid 点附近就越陡峭，像一座高耸的山峰。这代表了学生的**学习效率极高**。一旦他“开窍”，成绩就会在短时间内突飞猛进。参数 B 衡量的是**爬升效率**。一个高 B 值的方法，意味着它能非常高效地利用计算资源，在跨越中点后迅速接近其性能极限。

综合理解：

这张图告诉我们，评估一个RL训练过程，不能只看某个时间点的分数高低，而要用一个更全面的、动态的视角。S型曲线框架提供了一个“三维”的评估体系：

- **高度（A）**：决定了你最终能爬多高。
- **位置（Cmid）**：决定了你从哪里开始快速爬升。
- **坡度（B）**：决定了你爬升得有多快。

一个理想的RL配方，应该追求**高 A 值、低 Cmid 值和高 B 值**。但在现实中，往往需要在这些参数之间进行权衡。例如，某个方法可能 A 值极高，但 Cmid 值也很大，意味着需要极大的前期投入才能看到效果（“大器晚成”）。另一个方法可能 Cmid 很小，B 很大，早期进步神速，但 A 值不高，很快就到达瓶颈。

图3和公式(1)共同构成了这篇论文的科学基石。正是有了这个强大的分析工具，作者们才能在后续章节中，对各种复杂的RL技术进行庖丁解牛式的分析，清晰地判断出每项技术到底是在“提升潜力”、“提高起步效率”还是“加快爬升速度”。

3. 强化学习规模化的实证研究

原文翻译

在本节中，我们使用一个8B稠密模型在可验证的数学问题上进行RL实验。使用第2节中描述的设置，我们根据其可预测的计算规模化行为，即渐近性能（A）和计算效率（B），研究了几个设计轴，如图3所示。

我们将实验分为三个阶段：我们首先在基线上对设计选择进行消融研究，实验规模为3.5k到4k GPU小时，因为一些实验选择在此规模之外会变得不稳定（附录A.15）。当一个设计更改被证明是稳定的，我们就将其训练更长时间。然后，我们将最佳选择组合成ScaleRL，并在第4节中进行16k GPU小时的留一法（LOO）实验。在这里，我们通过在前8k GPU小时上进行拟合，并外推剩余的运行来评估可预测性。最后，为了展示ScaleRL的可预测规模化，我们还在第5节中考虑了更大批量大小、混合专家模型、多任务（数学和代码）以及更长序列长度的训练设置。

3.1 异步RL设置

我们首先研究异步离策略RL设置（Noukhovitch et al., 2024）的选择，因为它决定了训练的稳定性和效率，通常独立于所有其他设计选择。具体来说，我们考虑两种离策略学习方法：PPO-off-policy-k 和 PipelineRL-k。

PPO-off-policy-k 是异步RL的默认方法，之前已被Qwen3 (Yang et al., 2025) 和 ProRL (Liu et al., 2025a) 使用。在这种设置中，旧策略 $\pi_{\text{gen}}\theta_{\text{old}}$ 为一批B个提示生成推理轨迹。每个梯度更新处理一个大小为 B^ 的小批量提示，从而每个大批次产生 k=B/B^ 次梯度更新。在我们的实验中，我们固定 B^=48 个提示（每个有16个生成），并通过设置 B=k×48 来改变 $k \in \{1,8\}$。

PipelineRL-k 是Piche et al. (2025) 提出的最新方法，并被Magistral (Rastogi et al., 2025) 使用。在这种方案中，生成器以流式方式持续产生推理轨迹。每当训练器完成一次策略更新，新的参数就立即被推送到生成器，生成器使用更新后的权重继续生成，但使用的是来自旧策略的陈旧KV缓存。一旦生成了完整的轨迹批次，它就被传递给训练器进行下一次更新。在我们的设置中，我们引入了一个参数k：如果训练器领先生成器k步，它们就会等待。

深度解读

这一节开始进入论文的“实验”部分。作者们将像侦探一样，对影响RL训练效果的各种“嫌疑人”（设计选择）进行逐一排查。他们把这个排查过程精心设计成了三个阶段，体现了科学研究层层递进的严谨性。

三阶段实验设计

1. **第一阶段：小规模“海选”**。在3500-4000 GPU小时这个相对较小的计算规模上，对各种可能的设计选项进行广泛的初步测试。这个规模的选择很巧妙：它既足够大，能够看出不同方法的基本趋势；又足够小，能够控制成本，并且可以及时发现那些“不靠谱”的、容易导致训练崩溃的选项。这个阶段的目标是“去粗取精”，筛选出有潜力、表现稳定的候选技术。

2. **第二阶段：中等规模“复赛”与组合验证**。将第一阶段筛选出的“最佳”选项组合成最终的ScaleRL配方。然后，在一个更大的计算规模（16000 GPU小时）上，进行一种叫做“**留一法（Leave-One-Out, LOO）**”的消融实验。这种实验非常严谨：他们从完整的ScaleRL配方中，每次只“拿掉”一个组件，换回原来的基础版组件，然后观察性能变化。这就像测试一个冠军团队，想知道每个队员的重要性，就让他们轮流休战一场，看缺少谁对团队影响最大。这个阶段的目标是验证ScaleRL配方中的每一个组件都是不可或缺的，并且证明组合后的整体效果是“1+1>2”的。同时，他们还会在这里测试S型曲线的**预测能力**，即用前8000小时的数据预测后8000小时的表现。

3. **第三阶段：大规模“决赛”与泛化性检验**。将最终确定的ScaleRL配方，应用到更具挑战性的场景中，比如使用更大的模型（MoE）、处理更长的文本、应对多种任务（数学+代码）、采用更大的批量，并进行超大规模的训练（长达10万GPU小时）。这个阶段的目标是证明ScaleRL的**可扩展性**和**泛化性**，表明它不是一个只能在特定设置下工作的“温室花朵”，而是一个真正强大、通用的方法。

3.1 异步RL设置：选择高效的“流水线”

作者们排查的第一个“嫌疑人”是**异步离策略RL设置**。这是一个关于如何协调“生成器”（学生做题）和“训练器”（老师教学）工作流程的底层架构问题。它直接关系到整个训练系统的效率和稳定性。

> **离策略（Off-policy）** 的含义是，老师在更新教学方法时，使用的是学生用“旧”方法做出来的作业。这样做的好处是数据可以重复利用，效率高。这里的参数 k 代表了作业的“陈旧程度”，k 越大，作业越旧。

作者比较了两种主流的异步工作流：

1. **PPO-off-policy-k（分批同步模式）**：这是传统的方式。可以想象成一个学期分为几个阶段。在第一阶段，所有学生用第一版教材做作业。做完后，所有作业统一交给老师。老师批改完，更新出第二版教材。然后进入第二阶段，所有学生再用第二版教材做作业。这种模式的缺点是，在老师批改和更新教材时，所有学生都在“等待”，造成了GPU资源的闲置。

2. **PipelineRL-k（流水线模式）**：这是一种更现代、更高效的方式。想象成一个**流动的生产线**。学生们（生成器）源源不断地把做完的作业传递给老师（训练器）。老师这边也是，只要批改完一小部分作业，总结出一点心得，就立刻把这个“教学更新”传递给学生们。学生们拿到后，马上就在下一道题里用上这个最新的方法。整个过程无缝衔接，GPU的等待时间被大大缩短。

接下来的图4a将用数据告诉我们，这两种模式的效率差异有多大。

图4：(a) 比较异步离策略RL设置的“计算规模化” (b) PipelineRL的不同最大离策略度

原文翻译

图4：(a) 比较异步离策略RL设置的“计算规模化”。 我们只报告了拟合的S型曲线（公式1）的B（缩放指数）和A（渐近通过率）参数。PipelineRL-k的效率要高得多，并且在大计算量极限下略好一些。(b) PipelineRL的不同最大离策略度。

我们比较了这些方法在图4a中。PipelineRL和PPO-off-policy达到了相似的渐近性能A，但PipelineRL显著提高了计算效率B；因此更快地达到了天花板A。这是因为PipelineRL减少了训练过程中的空闲时间。这个选择用更少的token产生了可靠的增益，使得在较低的计算预算下进行更大范围的扫描成为可能。我们还改变了PipelineRL的最大离策略度，并发现k=8是最佳的，如图4b所示，我们在附录A.11中对此进行了进一步讨论。

深度解读

这张图是作者们对第一个设计轴——**异步RL设置**——进行的“海选”结果。它清晰地展示了不同的工作流对训练效率和最终性能的巨大影响。

(a) 子图：PPO-off-policy vs. PipelineRL

这张图直接比较了两种不同的异步协作模式。

- **横轴**：计算资源投入（千GPU小时）。
- **纵轴**：模型性能（通过率）。
- **三条曲线**：
    - **PPO-off-policy-1**（蓝色）：传统的、几乎是“在线”的PPO模式，数据陈旧度最低。
    - **PPO-off-policy-8**（橙色）：允许数据有一定陈旧度的PPO模式，效率稍高。
    - **PipelineRL-8-offpolicy**（绿色）：采用流水线工作模式，数据陈旧度与橙色曲线相当。

核心发现：

1. **潜力天花板（A）相似**：从图例中可以看到，这三种方法的渐近性能 A 值都非常接近，均为0.520。这意味着，无论采用哪种工作流，这个模型+基础算法组合的**潜力上限**基本是固定的。工作流本身并不能让模型变得“更聪明”。

2. **效率（B）天差地别**：真正的区别在于计算效率参数 B。

    - PPO-off-policy-1的 B 值为2.15。
    - PPO-off-policy-8的 B 值为2.68，略有提升，说明允许一定的离策略度有助于提高效率。
    - **PipelineRL-8的 B 值高达4.55**，几乎是传统PPO模式的两倍！

这意味着什么？

B 值代表了S型曲线中间部分的“陡峭程度”。一个两倍于对手的 B 值意味着，PipelineRL能以**快得多的速度**达到那个0.520的性能天花板。从图上看，绿色曲线的爬升速度明显快于蓝色和橙色曲线。在相同的计算资源下（例如4000 GPU小时），PipelineRL达到的性能显著更高。

作者解释了原因：PipelineRL通过其“流式”的特性，**极大地减少了GPU的空闲等待时间**，从而更高效地利用了每一分计算资源。这个发现至关重要，因为它意味着在不牺牲最终性能的前提下，可以通过优化工作流来**大幅缩短训练时间**或在相同时间内**达到更高性能**。这对于昂贵的LLM训练来说，是实实在在的降本增效。

(b) 子图：为PipelineRL寻找最佳的“陈旧度”

在确定了PipelineRL是更优越的框架后，作者们进一步对它的一个关键超参数k（最大离策略度，即数据陈旧度）进行了微调。

- **比较了k=4, 8, 12三种设置**。
- **核心发现**：
    - 当k从4增加到8时，效率 B 从4.14提升到4.55，性能天花板 A 保持在0.520。这说明在一定范围内，允许更高的数据陈旧度可以进一步提升效率。
    - 然而，当k继续增加到12时，效率 B 反而略有下降（4.22），更糟糕的是，**潜力天花板 A 显著下降到了0.490**。这表明，数据如果“太旧”，就会开始提供错误的或误导性的学习信号，从而损害模型的最终性能。

最终结论

通过图4的实验，作者们为他们的ScaleRL配方找到了第一个关键组件：**采用PipelineRL作为底层异步框架，并将离策略度设置为8**。这个选择在不牺牲甚至略微提升最终性能的前提下，实现了计算效率的最大化。这是一个典型的科学决策过程：先在宏观层面选择最优的框架（PipelineRL vs PPO），再在微观层面为这个框架找到最佳的参数配置（k=8）。

3.2 算法选择

原文翻译

基于以上结果，我们采用PipelineRL-8作为我们更新后的基线。然后我们研究了六个额外的算法轴：（a）损失聚合，（b）优势归一化，（c）精度修复，（d）数据课程，（e）批次定义，以及（f）损失类型。在第4节中，我们将最佳选项组合成一个统一的配方，称为ScaleRL（可扩展的RL），并在16,000 GPU小时的更大规模上进行留一法实验。

损失类型 我们将非对称DAPO损失（公式8）与两种最近提出的替代方案进行比较：GSPO (Zheng et al., 2025a) 和 CISPO (MiniMax et al., 2025; Yao et al., 2025)。

- **GSPO** 在序列级别应用重要性采样，而不是GRPO的token级别公式。具体来说，GSPO将token级别的IS比率（公式2）更改为序列级别的比率：ρi(θ)=πtrain(yi|x,θ)/πgen(yi|x,θold)。
- **CISPO** 简单地将截断的IS与香草策略梯度（Ionides, 2008）相结合，其中sg是停止梯度函数：

$$J_{\text{CISPO}}(\theta)=E_{x\sim D,\{y_i\}\sim\pi_{\text{gen}}(\cdot|x)}\frac{G}{\sum_{i=1}^{G}\sum_{t=1}^{|y_i|}}\text{sg}(\min(\rho_{i,t},\epsilon_{\max}))\hat{A}_i\log(\pi_{\text{train}}(y_{i,t}|x,y_{i,<t},\theta))$$
(4)

图5a显示，GSPO和CISPO都显著优于DAPO，大幅提高了渐近通过率A。CISPO表现出更长时间的近线性奖励增长，并且在训练后期略优于GSPO，因此我们选择CISPO作为我们的最佳损失类型。关于离策略损失类型及其超参数鲁棒性的进一步讨论在第4节和附录A.17中有详细说明。

LLM logits的FP32精度 生成器和训练器依赖于不同的核心来进行推理和训练，这导致它们的token概率存在微小的数值不匹配（He & Lab, 2025）。RL训练对这种差异高度敏感，因为它们直接影响替代目标中的IS比率。MiniMax et al. (2025) 发现这些不匹配在语言模型头部尤其明显，并通过在生成器和训练器的头部都使用FP32计算来缓解这个问题。如图5b所示，精度修复将渐近性能A从0.52显著提高到0.61。鉴于这一明显的好处，我们将FP32精度修复纳入我们的ScaleRL配方中。

深度解读

在选定了高效的`PipelineRL-8`作为新的“底盘”后，作者们现在开始对“引擎”本身的关键部件——**算法核心**——进行细致的排查。这一节他们将测试一系列对最终性能有重大影响的算法选择。

1. 损失类型 (Loss Type)：决定学习目标的“公式”

“损失函数”是机器学习的核心，它定义了模型的“学习目标”。一个好的损失函数能够为模型提供清晰、稳定且有效的学习信号。作者们在这里比较了三种不同的损失函数，这是一场关乎模型**潜力上限**的关键对决。

- **DAPO（基准）**：这是他们之前基础版算法中使用的，代表了当时比较先进的水平。

- **GSPO**：一种新提出的方法。它与DAPO/GRPO的主要区别在于计算“修正系数”ρ（重要性采样比率）的方式。DAPO是在每个词（token）的层面上计算这个系数，而GSPO则是在整个句子（sequence）的层面上计算。这两种方式各有理论上的优劣，需要通过实验来验证。

- **CISPO**：另一种新方法。它的数学形式与前两者有较大差异，更接近于一种被称为“香草策略梯度”的经典RL算法，但加入了“截断”重要性采样的技巧。这里的sg（stop-gradient，停止梯度）是一个技术细节，意味着在计算修正系数时，不让它影响梯度的方向，只让它影响梯度的大小，这通常有助于稳定训练。

接下来的图5a将揭示这场对决的结果。作者提前剧透：**GSPO和CISPO都大幅提升了潜力天花板A**，这意味着更换损失函数是一个能够让模型变得“更聪明”的根本性改进。

2. LLM logits的FP32精度：确保“语言”的统一

这是一个非常微妙但极其重要的技术细节，它揭示了大规模分布式训练中的一个常见陷阱。

- **问题所在**：还记得我们有“学生”（生成器）和“老师”（训练器）吗？它们运行在不同的GPU上，并且使用的计算库（kernel）也不同（一个为推理优化，一个为训练优化）。这会导致一个问题：即使它们使用的是完全相同的模型参数，对于同一个输入，它们计算出的每个词的输出概率（logits）可能会有极其微小的、因浮点数计算误差导致的差异。这就像学生和老师用的计算器品牌不同，算同一个复杂公式，结果在小数点后第8位上差了一点点。

- **为什么这在RL中是致命的？**：因为RL算法（特别是带重要性采样的）对这个概率比值$\rho非常敏感。\rho是新旧两个概率相除得到的。如果分子和分母本身就有微小的、不一致的误差，这个误差在相除后可能会被放大，导致\rho$值产生剧烈波动，从而给模型一个混乱、错误的学习信号，严重影响训练的稳定性和最终性能。

- **解决方案：FP32精度修复**：研究者发现，这个问题主要出在模型的最后一层，即“语言模型头部”（LM Head）。解决方案很简单：强制让生成器和训练器在计算这最后一层时，都使用精度更高、更统一的**FP32（32位浮点数）**格式，而不是为了速度而使用的BF16等低精度格式。这就像是给学生和老师统一发放了同一品牌、同一型号的计算器，确保他们之间不会再有“沟通误差”。

作者同样提前剧透了图5b的结果：这个看似微小的改动，**将潜力天花板A从0.52戏剧性地提升到了0.61**。这是一个惊人的提升，说明在RL这个“精细活”里，魔鬼真的藏在细节中。任何可能引入噪声和不确定性的因素，都可能成为限制模型潜力的瓶颈。

图5：(a) 比较流行的损失函数 (b) 在LM Head使用FP32精度修复

原文翻译

图5：(a) 比较流行的损失函数：DAPO (Yu et al., 2025), GSPO (Zheng et al., 2025a), 和 CISPO (MiniMax et al., 2025)。 我们发现与DAPO相比，CISPO/GSPO实现了更高的渐近奖励。(b) 在最后一层（LM head）使用FP32精度可以极大地提升渐近奖励。

深度解读

这张图展示了作者们在算法选择上两项最重大的发现，这两项发现都直接关联到如何**突破模型的性能天花板（参数A）**。

(a) 子图：损失函数"三国杀"

这张图是DAPO、GSPO和CISPO三种损失函数的正面对决。

- **DAPO（蓝色曲线）**：作为基准，它的性能曲线在达到大约0.52的通过率后就早早地进入了平台期。其潜力天花板 A 值为0.520。
- **GSPO（橙色曲线）**：相比DAPO有了巨大的飞跃。它的曲线持续上升，最终达到了一个高得多的平台，潜力天花板 A 值为0.590。
- **CISPO（绿色曲线）**：表现甚至比GSPO还要略胜一筹。它的上升势头最为持久，尤其是在训练后期，展现出"后劲十足"的特性，最终的潜力天花板 A 值达到了0.595。

核心结论：损失函数的选择对模型的最终能力有**决定性**影响。从DAPO换到CISPO或GSPO，不是一个简单的效率提升，而是**直接将模型的潜力上限提升了一个档次**。这就像给一个有潜力的学生换了一套更科学、更符合他认知规律的学习方法，让他能够突破瓶颈，达到以前无法企及的高度。作者们因此毫不犹豫地选择了表现最佳的**CISPO**作为ScaleRL配方的核心组件之一。

(b) 子图：精度修复的魔力

这张图展示了在模型头部（LM Head）使用FP32高精度计算所带来的惊人效果。

- **Default（蓝色曲线）**：这是不进行任何精度修复的基线情况，其潜力天花板 A 值和(a)图中的DAPO一样，是0.520。
- **fp32 precision fix（橙色曲线）**：在应用了FP32精度修复后，性能曲线一路高歌猛进，最终的潜力天花板 A 值**飙升至0.610**。

核心结论：这是一个教科书级别的案例，展示了在大规模AI训练中，看似微不足道的**数值稳定性**问题可以对最终结果产生多么巨大的影响。仅仅是通过统一生成器和训练器在关键部分的计算精度，消除那些微小的数值误差，就如同打通了模型学习的"任督二脉"，使其潜力得到了极大的释放。这个0.09的 A 值提升（从0.52到0.61）是本次研究中通过单项技术改进获得的最大收益之一。

综合洞见

图5的两个实验共同指向了一个深刻的道理：**在追求强化学习规模化的道路上，最重要的事情是找到那些能够直接提升性能天花板（A）的根本性改进**。

- **更换损失函数**是从**学习目标**的数学定义层面进行的根本性优化。
- **修复精度问题**是从**计算过程**的物理实现层面进行的根本性优化。

这两项改进都触及了学习过程的核心，因此它们带来的回报也是最丰厚的。相比之下，我们将在后续看到，很多其他的技术调整，更多的是在“术”的层面进行优化，它们能提高效率（改变 B 或 Cmid），但无法像这两项技术一样，在“道”的层面提升模型的最终潜能。这为ScaleRL配方的构建奠定了坚实的基础。

原文翻译

损失聚合 我们评估了三种聚合RL损失的策略：（a）样本平均，其中每个轨迹贡献相等（如GRPO，附录A.2）。（b）提示平均，其中每个提示贡献相等（如DAPO，附录A.2）。（c）Token平均，其中批次中所有的token损失直接被平均，没有中间分组。比较结果显示在附录A.9（图14a）中。我们发现提示平均实现了最高的渐近性能，因此我们将这个选择用于ScaleRL。

优势归一化 我们比较了三种优势归一化的变体：（a）提示级别，其中优势由来自同一提示的轨迹的奖励标准差进行归一化（如GRPO，附录A.2）。（b）批次级别，其中优势由批次中所有生成的标准差进行归一化，如Hu et al. (2025a); Rastogi et al. (2025) 所用。（c）无归一化，其中优势计算为原始奖励减去提示生成的平均奖励，没有方差缩放（如Dr. GRPO (Liu et al., 2025b) 中所提议）。比较图显示在附录A.9（图14b）中，所有三种方法被观察到产生相似的性能。因此，我们采用批次级别归一化，因为它在理论上是合理的并且略好一些。这个选择在第4节的留一法实验中在更大规模上得到了进一步证实。

零方差过滤 在每个批次内，一些提示在其所有生成中产生相同的奖励。这些“零方差”提示具有零优势，因此贡献零策略梯度。默认的基线将这些提示包含在损失计算中，但尚不清楚它们是否应被包含在有效批次中。为了测试这一点，我们比较了默认设置与一个有效批次方法，其中只有具有非零方差的提示被包含在损失计算中，如Seed et al. (2025) 所做。注意，零方差过滤不同于DAPO中的动态采样（Yu et al., 2025）。前者仅仅是丢弃提示，而后者则重采样更多提示直到批次被填满。我们在图6a中显示，使用有效批次在渐近性能上表现更好；我们在我们的ScaleRL配方中采用了它。

自适应提示过滤 已经有许多数据课程策略被提出来用于RL训练以提高样本效率（An et al., 2025; Zhang et al., 2025b; Zheng et al., 2025b）。这里我们评估一个简单的变体，由An et al. (2025) 引入，其关键观察是，一旦一个提示对于一个策略来说变得太容易，它通常会保持容易。由于这些提示消耗了一些计算但不再贡献有用的梯度信号（第3.2节），最好将它们从未来的训练中排除。我们通过维护一个通过率历史记录，并永久地从后续轮次中移除任何通过率≥ 0.9的提示来实现这一点——我们称之为无正样本重采样。在图6b中，我们将这个课程与默认设置进行比较，在默认设置中，所有提示在整个训练过程中都被均匀地重采样。我们看到该课程提高了可扩展性和渐近奖励A。

深度解读

在确定了损失函数（CISPO）和精度修复（FP32）这两个“大杀器”之后，作者们继续对一系列更为精细的“微操”技术进行排查。这些技术虽然影响可能不如前两者那么颠覆性，但它们的累积效应同样至关重要。

1. 损失聚合 (Loss Aggregation)：如何计算“班级平均分”？

当老师批改完一大堆作业（一个batch）后，需要算一个总的“教学效果得分”来指导下一步教学。如何计算这个总分，就是损失聚合要解决的问题。

- **Token平均**：把所有作业里所有字的对错得分加起来求平均。这最简单，但可能会被特别长的作业主导。
- **样本平均 (Sample average)**：先算每份作业的平均分，再把所有作业的平均分加起来求平均。每份作业（completion）的权重一样。
- **提示平均 (Prompt average)**：对于同一个问题（prompt），可能生成了16份不同的作业。先把这16份作业的得分汇总，算这个问题的“综合得分”，再把所有问题的“综合得分”加起来求平均。每个问题（prompt）的权重一样。
- **结论**：实验发现（附录图14a），**提示平均**能达到最高的潜力天花板A。因此，ScaleRL采纳了它。这可能是因为它保证了训练的焦点在问题本身，而不会被某个问题的某次特别好或坏的生成所过度影响。

2. 优势归一化 (Advantage Normalization)：如何设定“奖惩标准”？

“优势”衡量了一次尝试比平均水平好多少。归一化则是决定这个“好多少”的衡量标尺。

- **提示级别**：在同一个问题的16份作业内部，计算一个标准差，用它来归一化优势。奖惩标准只在“小组内”有效。
- **批次级别**：在整个批次的所有768份作业中，计算一个统一的标准差。奖惩标准在“全年级”统一。
- **无归一化**：只减去平均值，不做标准差缩放。
- **结论**：实验发现（附录图14b），这三种方法最终性能**差不多**。作者们选择了**批次级别**，因为它理论上更合理（在更大的样本上估计方差更稳定）并且在实验中表现略好。这体现了科学决策中的一个原则：当效果相近时，选择理论上更站得住脚的那个。

3. 零方差过滤 (Zero-Variance Filtering)：是否要理会“没有信息量”的数据？

有时候，对于一个问题，模型生成的16个答案可能**全对**或**全错**。在这种情况下，所有答案的奖励都一样，计算出的“优势”全都是零。这意味着，从这个数据中学不到任何东西（梯度为零）。

- **问题**：要不要把这些“零信息量”的数据从损失计算中剔除？
- **结论**：图6a将证明，**剔除这些数据（effective batch）效果更好**，能达到更高的潜力天花板A。这很直观：与其浪费计算资源去处理那些学不到任何东西的数据，不如把计算力集中在那些“有对有错”、能提供丰富学习信号的数据上。

4. 自适应提示过滤 (Adaptive Prompt Filtering)：要不要反复做“已经会了的”题？

这是一种“**数据课程**”（Data Curriculum）或“**因材施教**”的思想。一个好老师不会让一个已经精通一元一次方程的学生天天做“2x=4”这种题。

- **方法**：作者们采用了一种简单的策略，叫做“**无正样本重采样**”（No-Positive-Resampling）。他们会追踪模型对每个问题的历史解答正确率。一旦某个问题的正确率高到90%以上，就认为模型已经“掌握”了这个问题，并把它从后续的训练池中**永久移除**。
- **结论**：图6b将证明，这种简单的“剔除已掌握题目”的策略，能够**提高可扩展性和潜力天花板A**。这说明，让模型把宝贵的“注意力”（计算资源）集中在那些它还没有完全掌握的、有挑战性的问题上，是更高效的学习方式。

通过这一系列细致的实验，ScaleRL配方的“配料表”变得越来越清晰和完整。

图6：(a) “零”方差过滤 (b) 自适应提示采样：无正样本重采样 vs 均匀采样

原文翻译

图6：(a) “零”方差过滤： 我们过滤掉批次中“零”方差（准确率为0或1）的样本，因为它们贡献零策略梯度，并发现它能实现更高的渐近性能。**(b) 自适应提示采样：** 在后续轮次中过滤掉通过率≥ 0.9的提示，能带来更高的渐近性能。

深度解读

这张图展示了两种数据处理策略的对比结果，它们都旨在让模型的学习过程“更聪明”、更高效。

(a) 子图：“零”方差过滤

这张图比较了两种处理“零方差”数据的方式。

- **batch（蓝色曲线）**：这是基线方法，即不过滤任何数据，将批次中所有样本（包括那些全对或全错的）都用于计算损失。它的潜力天花板 A 值为0.520。
- **effec_batch（橙色曲线）**：这是“有效批次”方法，即在计算损失前，先将那些“零方差”的提示（prompt）过滤掉。它的潜力天花板 A 值提升到了0.540。

核心结论：过滤掉无效的学习信号是有益的。当模型对某个问题的所有尝试结果都一样时，它无法从中判断出哪些行为更好、哪些更差，因此无法进行有效的学习。将这些“无效信息”剔除，可以让模型更专注于那些能够提供明确“好坏”对比的、信息量更丰富的数据，从而略微提升了其最终能达到的性能上限。这是一个“积小胜为大胜”的例子。

(b) 子图：自适应提示采样（数据课程）

这张图比较了两种抽取训练题目的策略。

- **Uniform-Sampling（蓝色曲线）**：这是基线方法，即在每一轮训练中，都从整个题库里均匀、随机地抽题。这意味着，即使模型已经完全掌握了某道题，它在后面还是有同样的机会被抽到。它的潜力天花板 A 值为0.520。

- **No-Positive-Resampling（橙色曲线）**：这是“数据课程”方法，即一旦模型对某道题的解答正确率超过90%，就把它从题库中“拉黑”，不再进行训练。它的潜力天花板 A 值提升到了0.535。

核心结论：让模型“温故而知新”固然重要，但更重要的是把时间花在“攻克难关”上。当模型已经对某些知识点掌握得非常牢固时，再反复练习只会带来边际效益递减。通过动态地将这些“已掌握”的题目移出训练集，可以迫使模型将计算资源和“注意力”集中在那些更具挑战性、更能促进其能力增长的难题上。这种“因材施教”的策略，同样有效地提升了模型的潜力天花板。

综合洞见

图6的两个实验都体现了一个共同的原则：**提升学习效率和效果的关键在于提升“学习信号”的质量**。

- **零方差过滤**是通过**剔除“无信号”数据**来提升信噪比。
- **自适应提示过滤**是通过**动态聚焦于“强信号”数据**（即难题）来提升学习效率。

这些看似细微的数据处理技巧，共同作用，为ScaleRL配方贡献了宝贵的性能提升。它们证明了，一个成功的RL配方，不仅需要强大的核心算法（如CISPO），还需要一系列与之配套的、精细化的数据处理和训练策略。这就像一个顶尖运动员，不仅要有天赋和好的训练方法，还需要科学的营养和康复计划来辅助，才能达到最佳状态。

4. ScaleRL：有效且可预测地扩展RL计算

原文翻译

从上述研究的设计轴中，我们将表现最佳的设置整合到一个单一的配方中，我们称之为ScaleRL（可扩展的RL）。ScaleRL是一个异步RL配方，它使用具有8步离策略度的PipelineRL，基于中断的长度控制进行截断，对logits使用FP32计算，并优化$\mathcal{J}_{ScaleRL}(\theta)$损失。该损失结合了提示级损失聚合、批次级优势归一化、截断重要性采样REINFORCE损失（CISPO）、零方差过滤和无正样本重采样：

JScaleRL(θ)=E{xi},{yi}⊆S∼ED;{x}∑g=1G|yg|1i=1∑Gt=1∑|yi|sg(min(ρi,t,ϵ))A^inorm logπtrainθ(yi,t)
其中 ρi,t=πgenθold(yi,t)πtrainθ(yi,t), A^inorm=A^i/A^std, 0<mean({rj}j=1G)<1, pass_rate(x)<0.9

其中sg是停止梯度函数，$\hat{A}_{std}$是批次中所有优势$\hat{A}_i$的标准差，pass_rate(x)表示提示x的历史通过率。对于强制中断，我们使用结束思考的短语：“好了，时间到了。让我停止思考，现在构思最终答案。”。

留一法（LOO）消融 为了验证这些选择在组合时仍然是最佳的，我们进行了留一法（LOO）实验：从ScaleRL开始，我们一次将一个轴恢复到其在第2节中的基线对应物。这确保了每个设计决策即使在所有其他决策存在的情况下也能做出积极贡献。图7报告了这些实验，每个实验都扩展到16k GPU小时。在所有轴上，ScaleRL始终是最高效的配置，在

渐近奖励或计算效率上都略微优于LOO变体（请参考图7表格的最后一列）。由于大多数LOO变体达到了相似的渐近通过率，我们将S型拟合转换为幂律拟合，以通过斜率B突出效率差异（细节在图7中）。具体来说，我们对所有运行的渐近奖励A取平均值，用这个固定的A重新拟合曲线，然后在图7中比较斜率（衡量效率）。相应的未转换的通过率与计算曲线在附录A.2中提供。

深度解读

这一节是整篇论文的“成果汇总”和“压力测试”环节。作者们在系统性地测试了所有“配料”之后，终于正式公布了他们的“冠军配方”——**ScaleRL**，并用最严苛的科学方法之一来验证其设计的合理性。

ScaleRL：一个集大成的“冠军配方”

作者首先清晰地列出了ScaleRL的完整“配料表”。这个配方不是凭空创造的，而是前面所有实验中胜出者的集合，是系统性研究的智慧结晶：

- **底层框架**：`PipelineRL` (k=8) —— 最高效的异步工作流。
- **核心算法**：`CISPO` 损失函数 —— 潜力天花板最高的学习目标。
- **数值稳定性**：`FP32` 精度用于logits —— 消除计算误差，大幅提升潜力。
- **损失计算方式**：提示级损失聚合 + 批次级优势归一化 —— 理论更合理且效果最好的组合。
- **数据筛选策略**：零方差过滤 + 无正样本重采样 —— 提升学习信号质量，聚焦于有价值的数据。
- **长度控制**：强制中断 —— 简单有效的防止模型“失控”的机制。

这个最终的损失函数公式 JScaleRL(θ) 看似复杂，但实际上就是把上述所有优胜组件的数学思想融合在了一起。例如，公式里的 pass_rate(x)<0.9 就是“无正样本重采样”的数学表达，而 0<mean({rj}j=1G)<1 则对应着“零方差过滤”（均值不为0或1意味着并非全对或全错）。

留一法（Leave-One-Out, LOO）消融：最严格的“团队贡献”检验

仅仅把最好的东西组合在一起，并不能保证最终结果就是最好的。有时不同的组件之间可能会有“化学冲突”。为了证明ScaleRL的成功是源于**所有组件的协同作用**，而非某个单一的“神奇”组件，作者们进行了LOO实验。

这个实验的设计思想非常巧妙和严谨：

1. **基准**：完整的、包含所有最佳组件的ScaleRL配方。

2. **实验组**：创建多个“残缺版”的ScaleRL。每个版本都只将**一个**组件换回原来的、效果较差的基线版本。例如，“LOO-dapo”版本就是把CISPO损失函数换回DAPO，而其他所有组件保持不变。“LOO-PPO-off-policy-8”版本就是把PipelineRL框架换回PPO框架。

3. **比较**：将所有“残缺版”的性能与完整的ScaleRL进行比较。

如果ScaleRL的设计是合理的，那么我们预期会看到：任何一个“残缺版”的性能都应该**不如**完整的ScaleRL。这就像一个冠军球队，让任何一个主力队员轮休，球队的整体实力都会有所下降。

接下来的图7将展示这场“压力测试”的结果。作者提前透露，ScaleRL在所有比较中都胜出了，证明了其配方中的每一个决策都是积极且必要的。这种严谨的消融研究，是顶级科研工作的重要标志，它确保了结论的可靠性，排除了偶然性。

图7：留一法（LOO）实验

原文翻译

图7：留一法（LOO）实验： 从ScaleRL开始，我们一次将一个设计选择恢复到其基线对应物并重新训练。大多数LOO变体达到了相似的渐近奖励，ScaleRL总体上略微胜出。这些方法的主要区别在于效率。为了突出这一点，我们将公式(1)重新排列为 F(Rc)=CB，其中 F(Rc)=CmidB/(Rc−R0A−R0−1)，并绘制log F(Rc) vs. log C。这种形式使得斜率B直接可见，表明ScaleRL实现了最高的计算效率。

方法	拟合的B (固定A=0.605)	原始拟合参数 (Cmid / B / A)
ScaleRL	**2.01**	2542 / 1.92 / 0.610
L00-prompt-lvl-adv-norm	1.82	2463 / 2.09 / 0.590
L00-length-penalty	1.75	2660 / 1.68 / 0.610
L00-batch	1.69	2503 / 1.84 / 0.595
L00-PPO-off-policy-8	1.89	3409 / 1.89 / 0.605
L00-no-fp32-precision-fix	1.97	2857 / 1.88 / 0.610
L00-sample-avg	1.84	2621 / 1.93 / 0.600
L00-uniform-sampling	1.62	2621 / 1.62 / 0.605
L00-dapo	1.83	3093 / 1.77 / 0.610

深度解读

这张图和附带的表格是ScaleRL配方合理性的最终判决书。它通过严谨的留一法（LOO）实验，雄辩地证明了ScaleRL的每一个组成部分都对其卓越的性能做出了不可或缺的贡献。

图表的可视化技巧

首先，需要理解作者在这里使用的一个巧妙的可视化技巧。他们发现，在LOO实验中，大多数变体的**潜力天花板A**都非常接近（都在0.605左右），主要的区别在于**效率**。然而，在原始的S型曲线中，效率（由B和Cmid共同决定）的差异不够直观。

为了让“效率”差异一目了然，他们对S型曲线公式进行了数学变换，变成了一个类似幂律的形式 F(Rc)=CB。在这个变换后的坐标系中（log-log坐标），**曲线的斜率直接等于效率指数B**。这样一来，我们只需要比较图中各条线的“陡峭程度”，就能立刻判断出哪种方法的效率最高。

核心观察：ScaleRL效率全面领先

从变换后的图中可以清晰地看到，代表**ScaleRL**的蓝色直线，其**斜率是最大的**。这意味着它的效率指数B最高，计算效率在所有变体中是最佳的。其他所有“残缺版”的ScaleRL（即LOO变体），其对应的直线都比蓝线更平缓，说明它们的效率都更低。

表格的量化分析

右侧的表格为我们提供了更精确的量化数据，让我们能看清每个组件的具体贡献。

- **第一列（Method）**：列出了完整的ScaleRL和各种“残缺版”的名称。例如，“L00-dapo”代表用DAPO替换了CISPO的版本。
- **第二列（fitted B w/ fixed A=0.605）**：这是与左图对应的关键数据。为了公平比较效率，作者们固定了一个平均的潜力天花板A（0.605），然后重新计算每种方法的效率指数B。
    - **ScaleRL的B值高达2.01**，是所有方法中最高的。
    - 我们来看几个典型的“残缺版”：
        - **L00-dapo**：把CISPO换回DAPO，B值从2.01降至1.83。
        - **L00-PPO-off-policy-8**：把PipelineRL换回PPO，B值从2.01降至1.89。
        - **L00-uniform-sampling**：去掉了“数据课程”，B值从2.01大幅降至1.62。
- **第三列（original fitted parameters）**：这是对每种方法独立拟合S型曲线得到的原始参数。我们可以看到，虽然A值都比较接近，但ScaleRL的A值（0.610）仍然是最高的之一，并且其B值（1.92）和Cmid值（2542）的组合也是最优的。

综合结论

图7的LOO实验有力地证明了：

1. **协同效应**：ScaleRL的成功并非依赖单一的“银弹”，而是源于多个经过精心选择的组件之间的**正向协同作用**。

2. **每个组件都不可或缺**：从配方中移除任何一个关键组件，都会导致性能（主要是效率）的明确下降。这证实了作者们在第3节中通过“海选”得出的每一个结论都是正确且必要的。

3. **效率是关键差异**：在所有组件都达到较高水平后，它们之间的主要差异体现在“效率”上。ScaleRL的组合不仅将潜力推向了极限，更重要的是，它构建了一条**通往这个极限的最快路径**。

这个实验为构建可扩展的RL系统提供了一个范本：首先通过广泛的实验找到能提升潜力天花板A的核心组件，然后通过细致的比较和组合，找到一系列能最大化效率B、最小化Cmid的辅助组件，最终通过严谨的LOO消融来验证整个系统的设计的合理性。

原文翻译

拟合规模曲线的误差范围 由于RL训练已知表现出高方差（Agarwal et al., 2021），我们使用三个独立的ScaleRL运行（图8a）来估计拟合的规模系数的可变性。观察到的渐近奖励和效率参数的方差作为我们的经验误差范围，用于确定两次不同运行的计算效率或渐近性能的变化是否具有统计学意义（Madaan et al., 2024）。

外推规模曲线 在我们所有的LOO实验以及独立的ScaleRL运行中，我们将S型曲线拟合到8000 GPU小时，并外推到16000 GPU小时，观察到预测曲线与训练点和扩展点都紧密对齐。这证明了ScaleRL和其他稳定、可扩展的配方在大规模RL训练下的稳定性和可预测性。

这些设计选择值得吗？ 在第3.2节中，某些设计选择改变了渐近性能，例如损失类型（图5a）和FP32精度（图5b）。然而，在我们对ScaleRL的LOO实验中（图7），这些组件单独看来似乎不那么关键（图中的最后一列）。这就提出了一个问题：某些设计选择是否可以安全地保留其“默认”值。

我们认为答案是否定的。即使一个选择在组合配方中看起来是多余的，它仍然可以提供在其他情况下可能变得决定性的稳定性或鲁棒性。例如，虽然FP32精度修复在用ScaleRL训练的稠密8B模型上差异不大（图7），但它通过减轻数值不稳定性，在GRPO/DAPO风格的损失中提供了巨大的增益。这表明其好处超出了我们研究的特定ScaleRL配置。为了进一步测试这一点，我们在Scout 17B×16 MoE上进行了一次留一法实验，并观察到FP32精度提高了整体可扩展性（图8b）。

类似的情况也出现在损失类型上。在图7中，恢复到DAPO在ScaleRL内产生了与CISPO相似的渐近性能。尽管如此，正如我们在附录A.17中讨论的，CISPO对IS裁剪参数ϵ_{max}的选择明显更鲁棒，减少了训练对超参数调整的敏感性。此外，它也比DAPO更有效率，正如在LOO实验中看到的（B=2.01 vs B=1.77）。这证明了即使一个精心调整的DAPO变体可以在渐近性能上表现相似，也应优先选择CISPO。

总之，即使个别设计选择在组合配方中显得多余，它们通常会以一种可以跨模型和设置泛化的方式增强训练的稳定性、鲁棒性或效率。ScaleRL保留这些组件不仅仅是为了在特定配置中获得边际收益，而是因为它们解决了在各种强化学习机制中反复出现的不稳定性和方差来源。

深度解读

这一部分作者们深入探讨了两个关于他们研究方法论的关键问题：**结果的可靠性**和**设计的必要性**。这体现了顶级科学研究的自我审视和批判性思维。

1. 结果的可靠性：误差范围与可预测性

- **高方差问题**：作者们坦诚，强化学习训练本身就具有很高的“随机性”或“方差”。即使使用完全相同的代码和设置，两次独立的训练运行结果也可能不完全一样。这就像同一个学生用同样的方法复习两次，模拟考的分数也可能会有波动。

- **建立误差范围**：为了解决这个问题，他们没有只做一次实验，而是将核心的ScaleRL配方独立运行了**三次**（见图8a）。通过观察这三次运行结果的差异，他们可以估算出S型曲线拟合参数（特别是A和B）的“正常波动范围”，即**误差范围**。这就像通过多次测量来确定一个物理量的平均值和不确定度。有了这个误差范围，他们就可以科学地判断：当比较两种方法时，它们性能上的差异是真实存在的，还是仅仅是随机波动造成的。

- **验证可预测性**：作者们再次强调，在所有的中等规模实验（16000 GPU小时）中，他们都成功地用前8000小时的数据预测了后8000小时的结果。这反复证明了他们提出的S型曲线框架的**预测能力**是真实可靠的，不是偶然现象。

2. 设计的必要性：“多余”组件的价值

这里作者提出了一个非常深刻的自我诘问：在图7的LOO实验中，我们看到像FP32精度修复、将损失函数从DAPO换成CISPO等，对最终的潜力天花板A影响似乎不大（A值都在0.610左右）。那么，这些组件在最终的ScaleRL配方中是不是“多余”的？我们能不能为了简化，把它们换回更简单的默认选项？

作者给出了一个坚定的否定回答，并从**稳定性、鲁棒性和泛化性**的角度，阐述了这些“看似多余”组件的深层价值。

- **情景依赖性与稳定性**：一个组件的价值不能只在“最佳组合”这个单一情景下去评判。

  > **FP32精度修复的例子**：在已经非常优化的ScaleRL配方中（使用了稳定的CISPO损失），FP32的作用可能不明显。但是，如果你的配方使用的是像DAPO/GRPO这样对数值误差更敏感的损失函数（如图5b所示），那么FP32就成了防止训练崩溃的“救命稻草”。因此，将FP32作为标配，是为了**增强整个配方的普适性和稳定性**，确保它在更广泛的条件下都能稳健工作。作者通过在更大的MoE模型上的实验（图8b）进一步证明，即使在不同模型上，FP32依然能带来好处。

- **鲁棒性与开发成本**：

  > **CISPO vs DAPO的例子**：虽然一个**经过精心调参**的DAPO可以在ScaleRL框架下达到和CISPO差不多的潜力A，但作者在附录中指出，DAPO对超参数（特别是裁剪系数ϵ）的设置**极其敏感**，调得稍有不对，性能就会大幅下降。而CISPO则对超参数不那么敏感，**鲁棒性**（robustness）更强。在实际工程中，一个鲁棒性强的算法远比一个需要精细调参的“脆弱”算法更有价值，因为它大大降低了开发和维护成本。此外，即使A值相似，CISPO的效率B（2.01）也显著高于DAPO（1.83），选择它仍然是更优的。

核心思想

这一部分的论述传达了一个关于系统工程和科学研究的重要思想：一个优秀的、可扩展的系统，其设计决策的依据不应仅仅是**在理想条件下的峰值性能**，还必须同等重视**在各种非理想条件下的稳定性、鲁棒性和泛化能力**。ScaleRL之所以强大，不仅在于其各组件在“顺风局”中的优异表现，更在于它们共同构建了一个能够抵御各种潜在风险（如数值不稳定、超参数敏感）的坚固系统，确保在通往10万GPU小时的漫长征途中，训练过程能够行稳致远。

图8：(a) 分析不同运行间的误差范围 (b) Scout上的FP32留一法实验

原文翻译

图8：(a) 规模拟合的方差。 我们训练了3次独立的ScaleRL运行来测量方差。我们观察到渐近性能A有± 0.02的误差范围。**(b) Scout上的FP32 LOO：** 比较在Scout上使用和不使用LM Head的FP32精度修复的ScaleRL。带有FP32修复的ScaleRL表现更好。

深度解读

这张图通过两个实验，进一步巩固了上一节中关于**可靠性**和**必要性**的论点。

(a) 子图：量化不确定性，建立误差范围

这张图展示了三次完全独立的ScaleRL运行结果。正如预期的那样，由于强化学习训练内在的随机性，三条曲线并非完全重合，而是略有差异。

- **观察**：三条曲线的走势非常相似，但最终拟合出的参数有细微差别。
  - ScaleRL-1: B=1.92, A=0.610
  - ScaleRL-2: B=1.97, A=0.610
  - ScaleRL-3: B=2.20, A=0.595
- **核心结论**：通过这三次独立重复实验，作者们可以给出一个具有统计意义的结论：对于ScaleRL这个配方，其潜力天花板A的拟合值大约在0.60附近，波动范围约为± 0.015。为了保守起见，他们建立了一个**±0.02的误差范围**。

- **科学意义**：这个误差范围的建立至关重要。它意味着，当作者们比较两种新方法时，如果它们的A值差异小于0.02，他们就不能断定一种方法比另一种更好，因为这种差异可能仅仅是随机噪声。只有当A值的差异**显著大于0.02**时，他们才能充满信心地宣称这是一个**有统计学意义**的改进。这正是严谨科学研究区别于随意比较的标志。

(b) 子图：在更大模型上验证组件的必要性

这张图回应了上一节的自我诘问：FP32精度修复在ScaleRL中是否多余？为了回答这个问题，他们将实验场景从8B稠密模型切换到了一个规模更大、结构更复杂的**Scout 17B×16 MoE模型**上，然后再次进行了FP32的留一法（LOO）实验。

- **观察**：
    - **ScaleRL-17Bx16 MoE（蓝色曲线）**：这是在Scout模型上使用完整ScaleRL配方（包含FP32修复）的结果。其潜力天花板 A 值为0.710。
    - **ScaleRL-17Bx16 MoE-no-fp32（橙色曲线）**：这是去掉了FP32修复的版本。其潜力天花板 A 值下降到了0.700。
- **核心结论**：虽然0.01的A值差距看起来不大（在± 0.02的误差范围内），但橙色曲线在整个训练过程中的表现都稳定地低于蓝色曲线。这表明，即使是在一个更强大的模型上，FP32精度修复依然能带来**稳定、正向的性能增益**。
- **深层含义**：这个实验证明了FP32修复的**泛化性**。它的好处并不仅限于某个特定的模型或算法组合，而是一种能够**跨模型、跨设置**提供稳定性和性能提升的通用实践。这就雄辩地回答了之前的疑问：将FP32这样的组件作为ScaleRL的标准配置是完全必要且明智的，因为它能确保配方在更广泛的应用场景中都具有鲁棒性和高性能。

综合洞见

图8通过两个看似简单的实验，展示了顶级AI研究的两个重要侧面：

1. **对不确定性的尊重和量化**：图(a)告诉我们，科学的结论不是建立在单次完美的实验上，而是建立在多次重复、承认并量化随机误差的基础上的。
2. **对结论泛化性的严格检验**：图(b)告诉我们，一个真正好的技术，其价值不应只在最初发现它的那个狭窄场景中体现，而应在更广泛、更具挑战性的新场景中得到反复验证。

5. 跨RL计算轴的可预测规模化回报

原文翻译

给定一个固定或增长的计算预算，哪个规模化旋钮——上下文长度、批量大小、每个提示的生成数和模型大小——能带来最可靠的性能增益，我们能多早预测到这种回报？

我们通过以下方式回答这个问题：（i）在训练早期为每个设置拟合饱和幂律（公式(1)）（精确地说，是目标预算的一半），（ii）外推到目标预算，以及（iii）扩展训练以验证预测。在下面的所有轴上，我们观察到清晰、可预测的拟合，其外推曲线与扩展轨迹对齐，反映了我们在100,000 GPU小时运行（图1）和跨配方比较（图2）中看到的行为。

模型规模 (MoE) ScaleRL在更大的模型上是否保持可预测和稳定？使用ScaleRL训练17B×16的Llama-4 Scout MoE模型，表现出与8B模型相同的可预测规模化行为，具有低截断率且没有不稳定性病理（附录A.15, A.17）。图1显示了训练曲线。扩展点与拟合曲线对齐，支持了我们配方的模型规模不变性。此外，更大的17B×16 MoE表现出比8B稠密模型高得多的渐近RL性能，仅用其RL训练计算的1/6就超越了8B的性能。

生成长度（上下文预算） 将生成长度从14k增加到32k tokens会减慢早期进展（较低的B和较高的Cmid），但持续提升了拟合的渐近线（A），一旦提供了足够的计算，就会产生更高的最终性能（图9）。这验证了长上下文RL是一个提升天花板的旋钮，而不仅仅是一个效率权衡。从拟合中做的外推在训练被扩展时正确地预测了更高的32k-token轨迹。

全局批量大小（提示） 较小批量的运行在下游基准测试上显示出早期停滞，即使在分布内验证性能持续提高。更大的批量可靠地提高了渐近线，并避免了我们在较小批量运行中观察到的下游停滞。图10a在中等规模下显示了相同的定性模式：小批量可能早期看起来更好，但随着计算量的增长而被超越。在我们图1中最大的数学运行中，转向2048个提示的批量大小既稳定了训练，又产生了一个从高达50k GPU小时外推到最终100k点的拟合。

每个提示的生成数（固定总批量） 对于一个固定的总批量，是分配更多的提示更好，还是每个提示更多的生成更好？扫描每个提示的生成数8,16,24,32并调整提示数以保持总批量固定，拟合的规模曲线基本保持不变（附录A.13），这表明，在中等批量下，这种分配对于A和B都是一个二阶选择。更清晰的差异可能会在更大的批量（例如，2k+）下出现，我们留给未来的工作。

深度解读

这一节是整篇论文的“**实践应用指南**”。在证明了ScaleRL配方和S型曲线框架的有效性之后，作者们开始回答一个所有AI工程师都最关心的问题：**如果你有一笔固定的计算预算，应该如何分配它才能获得最大的性能回报？** 就像一个项目经理拿到一笔预算，需要决定是该多招几个初级工程师，还是少招几个但都是资深专家。

作者们考察了几个最关键的“**规模化旋钮**”（scaling knob），即可以投入计算资源来提升性能的维度。对于每一个旋钮，他们都做了同样严谨的验证：先用一半的预算进行训练并拟合S型曲线，然后用这个曲线去预测跑完整个预算后的最终性能，最后再实际跑完来验证预测的准确性。结果是，在所有维度上，预测都非常成功。

以下是他们对每个“旋钮”的洞察：

1. 模型规模 (Model Scale)

- **操作**：从8B参数的稠密模型换成一个更大、更先进的17B×16混合专家（MoE）模型。

- **结论**：这是**最有效**的提升性能天花板（A）的方法。如图1所示，更大的MoE模型不仅最终性能远超8B模型，而且学习效率极高，只用了8B模型1/6的计算量就达到了8B模型的最终性能。

- **启示**：在拥有了稳定可扩展的训练方法（如ScaleRL）之后，**模型自身的基础素质（规模和架构）是决定其潜力上限的最关键因素**。“好马配好鞍”，ScaleRL这个“好鞍”能让“好马”（大模型）跑得更快、更远。

2. 生成长度 (Generation Length)

- **操作**：将模型处理的文本长度（思考+解答）从1.4万tokens增加到3.2万tokens。

- **结论**：这是一个典型的“**先苦后甜**”的投资。图9将显示，使用更长的上下文进行训练，在早期会更慢、更低效（因为处理长文本更耗时），但它的**潜力天花板A会更高**。一旦计算量足够，长上下文训练的模型最终性能会反超短上下文训练的模型。

- **启示**：训练模型处理长文本，不仅仅是为了让它能读更长的文章，更重要的是，这个过程本身就能**提升模型的推理能力上限**。这就像让学生从做短小的填空题，转向解决需要长篇论证的复杂应用题，虽然上手难，但最终能培养出更强的综合能力。

3. 全局批量大小 (Global Batch Size)

- **操作**：增加每次参数更新所使用的样本数量（从512增加到768，再到2048）。

- **结论**：这同样是一个“**先慢后高**”的权衡。图10将显示，使用更大的批量进行训练，早期进展会更慢（因为处理一个大批次需要更长时间），但它能**可靠地提升潜力天花板A**，并且能**避免在下游任务上过早停滞**。

- **启示**：更大的批量意味着模型每次更新都是基于一个更广泛、更多样化的数据样本，这使得学习到的知识更具普适性，梯度更新方向更稳定。这就像一个政策制定者，是听取100个市民的意见，还是听取10000个市民的意见。听取更多人的意见虽然决策过程更慢，但最终制定的政策可能会更稳健、更有效。

4. 每个提示的生成数 (Generations per Prompt)

- **操作**：在总批量大小固定的前提下，调整是“多看题”（更多prompt）还是“一题多解”（每个prompt更多generation）。

- **结论**：在中等批量规模下，这个选择对最终性能（A和B）**影响不大**，是一个“二阶”因素。

- **启示**：这表明，在一定的范围内，模型学习的“信息总量”是关键，至于这些信息是以“广度”（多题）还是“深度”（多解）的形式呈现，区别不大。作者也指出，在超大批量下，情况可能会有所不同。

总结

这一系列实验为大规模RL训练提供了极其宝贵的实践指导。它告诉我们，为了追求更高的性能上限，最值得投资的“旋钮”是**模型规模**、**生成长度**和**批量大小**，尽管后两者可能会牺牲一些早期效率。而ScaleRL和S型曲线框架，正是确保这些“长期投资”能够稳定、可预测地获得回报的科学保障。

图9：扩展RL生成长度

原文翻译

图9：扩展RL生成长度。 虽然长上下文RL最初效率较低，但它最终会超越短上下文运行的性能。这一趋势在独立同分布（iid）验证集（左）和下游评估（右）上都观察到了。

深度解读

这张图直观地展示了投入计算资源于“**生成长度**”这个维度所带来的回报，并完美诠释了“**先苦后甜**”的投资模式。

(a) 子图：验证集上的表现

这张图比较了两种设置在“闭卷考试”（iid验证集）上的表现。

- **ScaleRL-14k（蓝色曲线）**：使用1.4万token的生成长度进行训练。
- **ScaleRL-34k（橙色曲线）**：使用3.4万token的生成长度进行训练（原文为34k，应为32k之误）。
- **观察**：
    - **早期（< 8000 GPU小时）**：蓝色曲线（14k）始终在橙色曲线（34k）之上。这是因为处理更长的文本需要更多的计算时间，所以在相同的GPU小时内，34k设置完成的训练步骤更少，学习进展自然更慢。从S型曲线参数看，它的 Cmid 值会更大，B值会更小。
    - **后期（> 8000 GPU小时）**：情况发生了反转。橙色曲线追上并反超了蓝色曲线，并朝着一个明显更高的性能水平前进。
- **核心结论**：拟合出的S型曲线参数证实了这一点。虽然14k设置的效率指数B（1.92）略高于34k设置（1.86），但34k设置的**潜力天花板A（0.650）显著高于14k设置（0.610）**。这0.04的A值差距，正是后期性能反超的根本原因。

(b) 子图：下游任务上的表现

这张图展示了两种设置在“开卷应用题”（AIME-24数学竞赛）上的泛化能力。

- **观察**：其趋势与(a)子图惊人地一致。在早期，14k设置表现更好；但随着计算资源的持续投入，34k设置的优势逐渐显现，并最终实现了反超。
- **核心结论**：这表明，通过长上下文训练获得的更高能力，是**可以有效泛化**到真实世界复杂任务中的真实能力，而不仅仅是验证集上的“刷分”。

综合洞见

图9为AI研究者和工程师提供了一个关于资源分配的重要启示：

1. **长上下文训练是“潜力投资”，而非“效率投资”**。如果你追求的是在短期、小计算预算下获得最快的性能提升，那么使用较短的上下文可能更合适。但如果你拥有充足的计算资源，并且目标是冲击性能的极限，那么投资于更长的上下文训练，尽管初期进展缓慢，但最终会带来更高的回报。

2. **“先苦后甜”模式的可预测性**。最关键的是，S型曲线框架能够**提前预见**这种“先苦后甜”的模式。研究者可以在训练早期，通过拟合曲线发现34k设置虽然当前表现不佳，但其潜力天花板A更高。基于这个预测，他们可以充满信心地继续投入资源，等待后期性能反超的到来，而不是被早期的落后表现所迷惑而过早地终止实验。

3. **为什么长上下文能提升潜力？** 一个可能的解释是，处理和生成更长的、连贯的文本，迫使模型学习更复杂的依赖关系、更长远的逻辑链条和更高层次的抽象结构。这种“磨炼”从根本上提升了模型的推理和规划能力，从而抬高了它的性能上限。

图10：扩展RL批量大小

原文翻译

图10：扩展RL批量大小。 更大的批量大小在训练中较慢，但会稳定在更高的渐近线上。批量大小最初显示出相反的趋势，即较小的值在较低的计算预算下似乎更好，但在更大规模下达到更高的渐近性能。

深度解读

这张图展示了另一个关键“规模化旋钮”——**批量大小（Batch Size）**——对性能的影响。它再次印证了与生成长度类似的“**先慢后高**”模式，并揭示了批量大小对于模型泛化能力的重要性。

(a) 子图：验证集上的表现

这张图比较了三种不同批量大小在“闭卷考试”（iid验证集）上的表现。

- **ScaleRL-bs512（蓝色曲线）**：使用512的批量大小。
- **ScaleRL-bs768（橙色曲线）**：使用768的批量大小。
- **ScaleRL-bs2048（绿色曲线）**：使用2048的批量大小。
- **观察**：
    - **早期**：批量大小越小，曲线爬升得越快。512的设置在早期性能上领先于768和2048。这是因为在相同的计算时间内，小批量可以完成更多的更新步骤。
    - **中期和后期**：随着计算量的增加，趋势开始反转。更大批量的曲线逐渐追上并反超了小批量的曲线。最终，2048批量的绿色曲线达到了最高的性能水平。

- **核心结论**：S型曲线参数清晰地揭示了背后的原因。
    - **潜力天花板A**：随着批量大小的增加而**稳步提升**（0.605 -> 0.610 -> 0.645）。
    - **效率B**：随着批量大小的增加反而**略有下降**（1.77 -> 1.92 -> 1.70，这里的1.92略高是个小异常，但总体趋势是下降或持平）。
    - **中点Cmid**：随着批量大小的增加而**显著增大**（未在图中直接标出，但在附录表格中有数据），意味着需要更多的前期计算才能进入快速增长期。

(b) 子图：下游任务上的表现

这张图展示了不同批量大小在“开卷应用题”（AIME-24）上的泛化能力，其趋势与(a)子图完全一致。

综合洞见

图10的实验结果，尤其是与原文中“较小批量的运行在下游基准测试上显示出早期停滞”这句话相结合，为我们提供了关于批量大小作用的深刻理解：

1. **大批量是通往更高泛化能力的基石**。为什么更大的批量能带来更高的潜力天花板A？因为每一次参数更新都是基于一个更大、更多样化的数据样本。这使得模型学到的“知识”或“策略”更具普适性，不容易过拟合到训练数据中的某些特定模式。梯度更新的方向也更稳定、更准确，更能代表全局最优的方向。这就像制定政策时，听取更多元化的意见能制定出更稳健的政策一样。

2. **避免“在验证集上过拟合”的陷阱**。作者特别提到，小批量训练有时会出现一个奇怪的现象：在验证集（闭卷考试）上分数还在涨，但在下游任务（开卷应用题）上分数已经停滞不前了。这是一种“高分低能”的信号，说明模型可能开始过拟合验证集的分布了。而**使用更大的批量可以有效缓解这个问题**，确保模型在验证集上的性能提升能够真实地转化为下游任务的泛化能力提升。

3. **再次验证“先慢后高”的可预测性**。与生成长度一样，S型曲线框架再次成功地预测了这种“先慢后高”的趋势。研究者可以在早期就判断出，虽然2048批量的设置当前落后，但其更高的A值预示着它拥有更大的长远潜力，从而做出继续投资的正确决策。

总结来说，这项研究表明，在进行大规模RL训练时，**采用尽可能大的批量大小**是一个明智的“长期投资”策略。它可能会牺牲一些早期的训练速度，但换来的是更稳定的训练过程、更高的性能上限和更强的泛化能力。

图11：ScaleRL在数学和代码上可预测地扩展

原文翻译

图11：ScaleRL在数学和代码上可预测地扩展。 我们报告了在联合数学-代码RL运行中，代码和数学验证集的性能；同时以仅数学的ScaleRL运行作为参考。这些结果表明，我们的S型计算-性能关系在任务混合中仍然成立，并且ScaleRL的可扩展性可以泛化到单一领域训练之外。

深度解读

这张图将ScaleRL的适用范围从单一任务（数学）扩展到了**多任务学习（Multi-task Learning）的场景，即同时训练模型解决数学和编程两种不同类型的问题。这是对ScaleRL配方泛化能力**的一次重要检验。

实验设置

- **Math（蓝色曲线）**：这是基准实验，即只在数学数据集（Polaris-53k）上训练模型。这条曲线与我们之前看到的ScaleRL基线是相同的。
- **Multi-task, math（橙色曲线）**：这是一个新的实验。模型在一个混合了数学和编程问题的数据集上进行训练。这条橙色曲线代表了在这个多任务训练过程中，模型在**数学验证集**上的表现。
- **Multi-task, code（绿色曲线）**：同样是在这个多任务训练中，这条绿色曲线代表了模型在**代码验证集**上的表现。

核心观察与结论

1. **S型曲线在多任务场景下依然有效**：最重要的一点是，无论是橙色曲线（多任务中的数学能力）还是绿色曲线（多任务中的代码能力），它们都呈现出非常平滑、清晰的S型增长轨迹。这有力地证明了作者们提出的S型计算-性能框架**不是一个只适用于单一任务的特例**，而是一个具有普适性的、能够描述LLM在RL训练中能力增长的基本规律。
2. **ScaleRL配方的泛化性**：ScaleRL这个配方在更复杂的多任务环境中，依然表现出稳定、可预测的规模化特性。这说明ScaleRL的设计原则（如PipelineRL, CISPO, FP32等）是足够底层的、通用的，能够很好地适应不同类型任务的学习。
3. **任务间的潜在影响**：
    - 比较蓝色曲线（只学数学）和橙色曲线（同时学数学和代码）在数学上的表现。我们可以看到，两条曲线的最终潜力天花板A非常接近（蓝色0.610 vs 橙色0.595），在误差范围内几乎可以视为相同。这表明，**引入代码任务并没有损害模型在数学任务上的最终潜力**。
    - 橙色曲线的效率B（2.05）甚至略高于蓝色曲线（1.92），这可能暗示着学习编程和学习数学之间存在一些**正向的迁移**（positive transfer），例如，两者都依赖于逻辑推理和结构化思维，同时学习可能有助于强化这些底层能力。
    - 绿色曲线（代码能力）的潜力天花板A（0.615）也非常高，与数学能力相当，但其效率B（1.09）相对较低。这可能意味着，对于这个特定的模型和数据集，代码任务本身的学习难度更大，或者说模型需要更长的时间来“开窍”。

综合洞见

图11的实验极大地增强了这篇论文结论的说服力。它证明了ScaleRL和S型曲线框架的**鲁棒性和通用性**。

- 对于AI研究者来说，这意味着他们可以放心地将这套方法论应用到更复杂的、包含多种任务的训练场景中，去打造能力更全面的通用人工智能模型。
- 它也揭示了多任务学习的一个有趣现象：不同任务的学习曲线（潜力A和效率B）可能是不同的，但它们都遵循相同的基本规律。这为未来研究如何优化多任务学习