BenchEvolver

以解为中心的进化式前沿任务合成

作者之一 & 报告人：Hang Wu (ID: nike0good)　(拉格朗日具身·*Lagrange Embodied*)

作者：Yangzhen Wu, Aaron J. Li, Hang Wu, 等

15 分钟口头报告

说明：Hang Wu 是本工作的作者之一；因投稿时 OpenReview 账号仍在审核（pending）未及署名，将于后续（转投 / 正式版本）中补入作者名单。

UC Berkeley · Tsinghua IIIS · XJTU

ICML AI4Math Workshop ——*Poster*

一个温馨的小注脚……

作者名单 (Authors)

- Yangzhen Wu
- Aaron J. Li
- Wenjie Ma
- ……
- Hang Wu ← 我

拉格朗日具身

署名拟于后续版本补入

目录

研究动机：基准正在饱和

前沿模型已让基准饱和

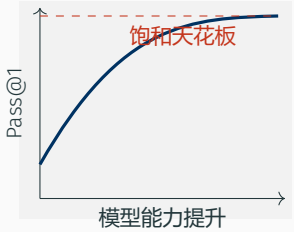
在 LiveCodeBench 上，最先进的模型：

- 在最新的简单子集上 Pass@1 超过 99%
- 平均 Pass@1 超过 90%

由此带来的问题：

- **失去区分度**：无法分辨强模型之间的差距
- **失去训练梯度**：几乎没有可学习的信号

这不止发生在代码领域——推理、科学、智能体任务同样随模型进步而失去区分度。



人工构造更难的基准既昂贵又难以规模化。

* LiveCodeBench：从 LeetCode / AtCoder / Codeforces 持续抓取的竞赛编程基准，按题目发布时间滚动更新以**防数据污染**；是本文进化的主要种子来源。

† Pass@1：模型**一次生成**、代码通过**全部隐藏测试**的比例（越高 = 越会做，100% = 饱和）。

缺口：自我挑战式的任务生成

现有的合成数据流程大多是非对称的：由强教师模型为弱学生模型构造训练/评测数据。

两个局限

1. 合成停留在**指令层面**——只改写表述，不改变底层解结构与真实难度。
2. 正确性依赖**更强的模型 / 人工审核** ⇒ 本质是蒸馏，而非自我提升。

我们真正需要的
同时满足**有效**、**可验证**、**难度可控**、**对生成者自身也困难**的任务——让模型暴露并修复自己的弱点。

核心思想：进化“解”，而非“题面”

以解为中心的进化

传统做法（题面优先）：

$$S' \longrightarrow \text{推断} C',\ T'$$

脆弱：当同一模型既出题又验证时，含糊的表述与隐含假设容易蒙混过关。

BenchEvolver（解优先）：

$$C \longrightarrow C' \longrightarrow I' = (S', C', T', E)$$

先变异**参考解**，再围绕它反推题面、样例与测试。

为什么有效

难度成为任务**计算本身**的属性，而非表面形式。

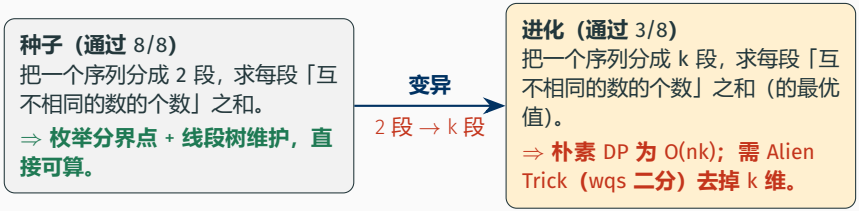
Proposer 被要求引入**主导性的算法跃迁**：

- 更强的渐进复杂度策略
- 更丰富的数据结构 / 状态维护
- 全新的数学重构
- 使父解捷径失效的自然约束

样例与测试输出均由执行 C' 生成——牢牢锚定在可执行语义之上。

任务四元组 $I = (S, C, T, E)$：S = 题面 · C = 参考解 · T = 隐藏测试集 · E = 执行框架（唯一与领域相关） · 上标 $'$ = 进化后版本。

一个具体的变异例子



- 题目骨架相同，只把「2 段」放宽成「k 段」。
- 「**枚举分界点 + 线段树**」的做法直接失效——必须换成 Alien Trick 这类更高级的优化。

* Alien Trick（又称 wqs 二分 / 拉格朗日优化）：把「恰好分 k 段」的硬约束换成**每段罚一个** λ，二分 λ 去掉 k 这一维（在最优值关于 k 凸时成立）。

BenchEvolver 框架

三条原则，三个模块

在解空间**生成** · 以独立检查**验证** · 以经验失败**筛选**

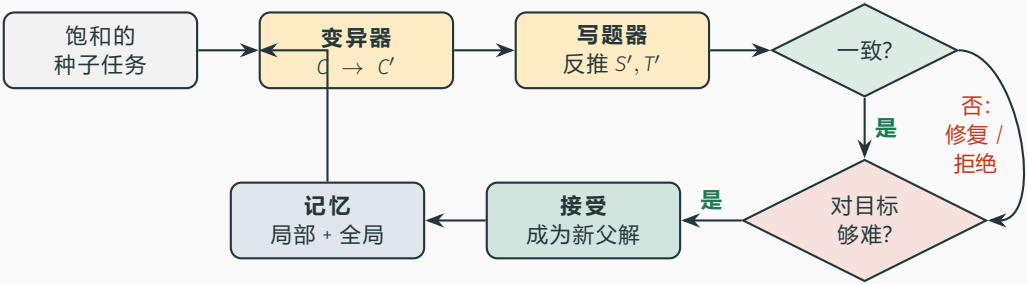
Proposer（出题者）
变异 $C \to C'$，反推 (S', T')；以历史反馈为条件，只负责提议，绝不自我接受。

Evaluator（评估者）
一致性检查 + 经验难度评估；有限预算内修复，否则拒绝。

Memory（记忆）
局部（单种子）+ 全局（跨种子）历史，引导搜索并强制多样性。

任务四元组 $I = (S, C, T, E)$：S= 题面、C= 参考解、T= 隐藏测试、E= 执行框架；
只有执行框架 E 与领域相关（标准输入输出 vs. 断言式测试）。

闭环流程



- **难度由行为定义**：每个目标模型尝试 k 次，通过全部隐藏测试才算成功。
- 只有当难度**超过种子**（可选地超过父解）时才接受。

如何保证有效性（无需更强的教师）

竞赛编程——暴力三角验证
交叉验证三个“观测视角不同”的证据源：

- 进化后的参考解 C'
- 仅凭题面写出的**暴力**求解器
- 仅凭题面产生的**公开输出**判定器

它们的分歧可精确定位问题出在参考解、暴力解还是题面本身。

科学计算——题面忠实性检查
仅从题面独立生成的一份解，跑候选测试：判断题面是否足以确定预期的计算。

过滤“虚假难度”：歧义、误导性 I/O、不自然的边界、换皮式近似重复。

实验结果

进化产出与有效性——LiveCodeBench-v6

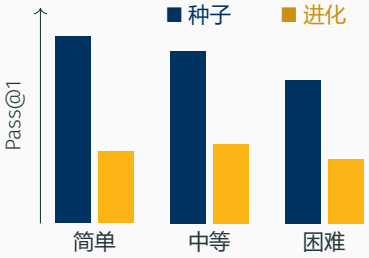
方法	进化模型	简单	中等	困难	有效性
题面为中心	GPT-5.4-mini	54.5%	47.8%	30.0%	79.3%
无记忆	GPT-5.4-mini	59.1%	65.2%	45.0%	86.2%
BENCHEVOLVER	GPT-5.4-mini	**77.3%**	**69.6%**	**60.0%**	**97.7%**
BENCHEVOLVER	Gemini-3-Flash	81.9%	60.9%	80.0%	89.9%
BENCHEVOLVER（前沿）	Gemini-3.1-Pro	80.0%	90.0%	60.0%	96.7%

- **解为中心** ≫ **题面为中心**：产出更高，且有效性 +18 个百分点。
- **记忆有用**：已接受的谱系与历史失败胜过独立的一步变异。
- 跨多个进化模型、轻量与前沿两个目标层级均成立。

进化任务确实更难

Pass@1 下降（种子 → 进化，均值 Δ，k=4）：

- GPT-5.4-mini：Δ = -0.265
- GPT-5.4：Δ = -0.282
- Gemini-3-Flash：Δ = −0.184
- Gemini-3.1-Pro：Δ = −0.159
- Claude Opus 4.6：Δ = -0.50 至 −0.58



在每个难度层级、每个进化模型上都一致下降。

关键性质

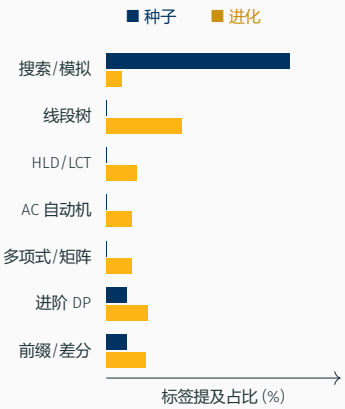
每个进化模型在自己生成的任务上都会掉分。这不是教师 → 学生，而是模型**暴露自身**弱点。

* Δ = 进化题相对种子题的 Pass@1 变化（每题 k=4 次尝试取平均）；负值 = 更难。

拓宽的是算法面，而非更长的题面

盲审人评——6 **位** CF **大师** / IOI / ICPC **级专家**，100 条谱系，207 道进化题：

- 种子被单一范式主导：**搜索/模拟** = 32.7%
- 进化题的分布扩散到**线段树、树链剖分/LCT、AC 自动机、多项式/矩阵、进阶** DP
- 不同算法类别：19 → 30
- 95.6% 的谱系至少引入 1 个新类别（平均约 2.54 个/谱系）



跨领域推广：科学计算 SciCode

目标层级	进化模型	进化种子占比	有效性
轻量	GPT-5.4-mini	88.9%	91.0%
轻量	Gemini-3-Flash	96.3%	98.2%
轻量	Claude-Sonnet-4.6	29.6%	90.0%
前沿	Gemini-3.1-Pro	60.7%	88.9%

- 无暴力判定器可用 ⇒ 改用题面忠实性检查 + 断言式测试。
- 依然产出**有效且显著更难**的科学计算任务。
- **以解为中心的进化并不局限于单一领域**——配上合适的验证协议即可迁移。

* SciCode：面向**科研的科学计算**基准，题目来自真实论文中的物理 / 数学 / 化学等计算，以**函数级 + 断言测试**评测（区别于 LCB 的标准输入输出竞赛题）。

LiveCodeBench-Plus 与自我提升

LiveCodeBench-Plus：让基准重新具备区分度

91 道题 = 64 道验证过的进化题（20 中等 + 44 困难）+ 27 道原始困难 LCB-v6。
经人工质量门槛（$\geq 3/5$）+ 难度窗口（Pass@1 $\in [0.05, 0.75]$）筛选。

模型	困难-种子	困难-进化	LiveCodeBench-Plus
GPT-5.5	97.1	62.3	62.6
GPT-5.4	94.8	49.7	54.1
Gemini-3.1-Pro	96.5	56.8	59.1
GPT-5.4-mini	79.7	21.7	29.6
DeepSeek-V4-Pro	83.7	23.2	27.5

- 困难子集：平均 Pass@1 87.0% → 45.7%（−41.3 点）；对每个模型都成立。
- 整体分布：27.5% –62.6%——区分度得以恢复。

* LiveCodeBench-Plus：本文用 BenchEvolver 产出的**难度升级版基准**——64 道进化题 + 27 道原始困难题，共 91

闭合回路：进化任务作为强化学习信号

设置： `gpt-oss-20b` 同时充当进化模型与目标模型。880 道 LCB v1–v5 种子 → 586 道进化题；GRPO 训练；留出集评测。

留出集	基座	仅种子	仅进化	种子 + 进化
LCB v6 Hard	40.0	+5.1	+7.6	+8.7
LCB-Pro Easy	64.6	+6.2	+7.2	+8.3
LCB-Evolved Med	30.4	+3.2	+7.8	+6.9

- **种子 + 进化最佳**：在公开留出集上，混合数据取得最优。
- 相比仅种子的 RL，额外增益高出 +70.7%（v6 Hard）与 +34.8%（Pro Easy）。
- 在独立生成的进化集（Gemini 产生）上迁移依然成立——并非简单的数据重叠。

* GRPO（Group Relative Policy Optimization）：一种**无需价值网络**的强化学习算法，用同题多次采样的**组内相对奖励**来更新策略；此处奖励来自代码是否通过测试。

结论

核心要点

1. **先进化计算。** 变异参考解，再反推题面与测试——让难度扎根于可执行语义。
2. **经验式、自我挑战的难度。** 以模型失败衡量难度，而非评委打分——进化任务连生成者自己都难倒。
3. **“活的”基准。** LiveCodeBench-Plus（91 题）恢复区分度：Pass@1 27.5%–62.6%。
4. **从挑战到能力。** 在进化任务上做 RL 能提升留出集表现——闭合自我提升回路。

未来工作

多轮自我博弈（提升后的模型成为下一代进化模型）；课程与多样性控制；随前沿模型共同演化的活基准。

谢谢！

BenchEvolver

把饱和的基准，变成前沿评测与可复用的训练信号。

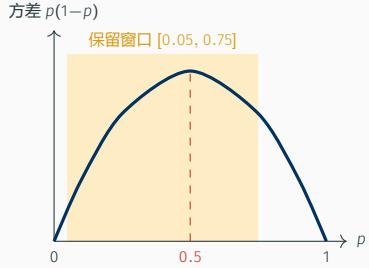
欢迎提问

Backup：为什么 LCB-Plus 能提高区分度?

1. **打下天花板**：困难子集 Pass@1 87% → 45.7%，把模型从「贴着 100%」拽回会分化的区间。
2. **难度窗口筛选** Pass@1 ∈ [0.05, 0.75]：主动扔掉太易 (> 0.75) 与太难 (< 0.05) 的题——这两类的区分度都 $= 0$。
3. **拓宽算法面**：类别 19 → 30，多维度考察 → 暴露不同模型的短板。
4. **新题防污染**：进化题没被背过，测的是真实能力，而非「记没记过」。

效果：原始困难题挤在 87–97% ⇒ LCB-Plus 摊开成 27.5–62.6%（拉开约 35 点）。

单题区分度最大 @ $p \approx 0.5$



单题相当于一次抛硬币：
$p \approx 0.5$ 时方差最大、最能拉开分数；
$p = 0$ 或 1 时人人同分、区分度为 0。