Evolvent AI

长程自进化：让一个自动科研系统越用越聪明

SPEAKER
胡梦康 · Founder & CEO

DATE
2026.7.3 · Shanghai

contact@evolvent.co

一群从 *顶尖实验室* 与 *世界一流学府* 走出来的人。

研究方向覆盖 LLM Agents · 强化学习 · 视觉语言模型 · 推理 · 长程规划 · Embodied AI —— 我们做过你们今天用的那一代 agent 系统，现在想把下一代做出来。

FRONTIER LABS · INDUSTRY

生产环境里跑过 Agent 基础设施

- Google — GOOGLE BRAIN
- Microsoft Research 微软亚洲研究院 — MSRA
- ByteDance — BYTEDANCE SEED
- Moonshot AI — MOONSHOT · KIMI
- ANT GROUP — ANT GROUP A-FU
- Alibaba — ALIBABA
- Tencent 腾讯 — TENCENT HUNYUAN

ACADEMIC BACKBONE

- CMU
- NUS National University of Singapore — NUS
- HKU
- SJTU

7
RESEARCH SCIENTISTS
核心科研团队

26K+
CITATIONS
Google Scholar 累计引用

150+
PAPERS
ICLR / NeurIPS / KDD / ACL …

38
H-INDEX · MAX
NeurIPS Oral · top 0.5%

AI Coworker 和 AI 助手，本质区别在哪？

数据范式从单轮，到一次会话，到 *跨日存活、持续学习* —— Agent 的形态，被它能积累什么决定。

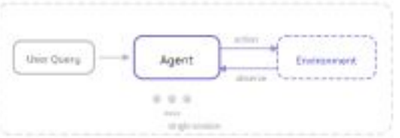
1 ERA 1 · CHATBOT

一次提问，一次回答

单轮 prompt → response。无状态、无环境、无记忆。关闭页面，Agent 就忘了你。

2 ERA 2 · SINGLE-SESSION AGENT

一次会话内，多步规划与工具调用



会话内多步：感知-规划-行动循环。但记忆随会话结束清零，下次见面又是陌生人。

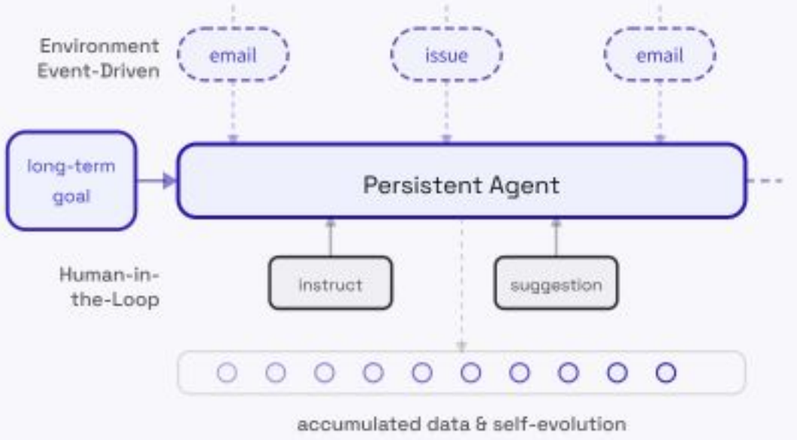
3 ERA 3 · PERSISTENT AGENT

跨日存活，从轨迹中学习 —— *AI Coworker 真正生活的地方。*

常驻、事件驱动。时间轴上每一步都依赖上一步。环境会变、目标会延展、人会插话。

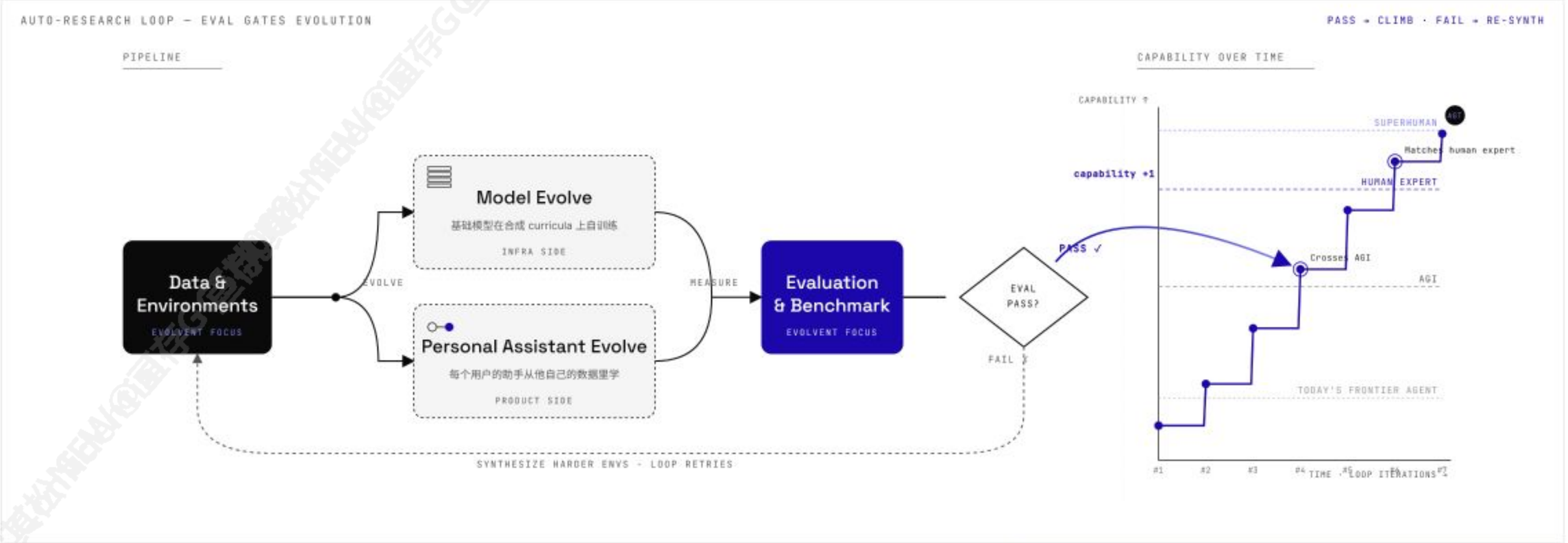
- 长程任务跨日推进
- 环境会动态演化
- 从轨迹中自进化
- 事件驱动触发
- 人机回路协同

• Evolvent's focus



通往 *独立行动者* 的最后一步 —— *自进化的 Agent*。

我们的所有研究和业务，都围绕这一张图展开 —— 一头生成环境与数据，一头测量与评估， 中间是模型和 personal assistant 各自的进化轨道。*评估通过 → 能力上升一阶；评估失败 → 合成更难的环境，再来一遍。*



两个端点

能力被 *生成* 的地方（环境与数据），与被 *验证* 的地方（评估） —— 是闭环的两个端点。

两条进化轨道

*模型*在 infra 侧进化，*个人助手*在 product 侧进化 —— 同一台引擎，两个面向。

终态 · AGENT AUTONOMY

每个节点自运行 —— Agent *生成*自己的环境、*构建*自己的 benchmark、*跑*自己的进化循环。

CHAPTER 1 · 五项开源工作

我们对 self-evolution 的 *研究* 思考。

围绕自进化闭环的五个端点 —— 评估、harness、环境、信任、技能扩展 —— 我们在一周内连续放出五项开源研究。

EVAL

ClawMark

多回合、多日、多模态。把"AI Coworker 能做几天的活儿"做成了可重复的考题。

HARNESS

BenchRouter

Router 而非 framework —— 让评估能跟得上模型两周一次的迭代节奏。

ENV

Terrarium

时间轴上的环境，每一步依赖上一步 —— persistent agent 的 RL/eval 数据基座。

TRUST

AuthBench

Agent 的权限边界 —— 让信任可以分级、可以验证、可以复利。

SKILL SCALE

Physics of Skill

一个反直觉发现：技能多反而更差 —— 我们提出执行回流的解法。

NEURIPS 2026 · BENCHMARK TRACK

ClawMark

面向"AI 同事"的多回合多日多模态基准

现有 agent 评测多在*单一静态会话*中打分；但真正的 AI 同事会跨多个工作日陪人干活，环境在背后持续演化，证据散落在文档、音视频和表格里。**ClawMark 把这套真实工作流第一次完整搬进可执行的基准里。**

设计承诺 · 01

多回合时间轴

跨工作日的连续协作

一个任务由 2–6 个回合组成，每个回合对应一个"工作日"。Agent 必须跨日推进进度，而不是单步交付。

设计承诺 · 02

动态环境

回合之间环境会被外部改写

Loud events 在唤醒消息中宣布；silent mutations 不通知就改服务状态。可靠的同事每次都得"重新看一眼"。

设计承诺 · 03

原始多模态证据

不预转录的真实证据

图片、扫描 PDF、音频、视频、表格按原样下发，模型自己用 whisper / ffmpeg / PyMuPDF 等工具解析。

100	13	1,537	5
任务 / TASKS	职业场景 / SCENARIOS	规则化检查器 / CHECKERS	有状态沙箱服务 / SERVICES

TASK DESIGN

任务在*多日时间轴*上展开，环境在两次回合之间被外部改写。

每个回合 = 一个"工作日"，agent 收到唤醒消息，对五个有状态的沙箱服务进行操作。回合之间环境会发生变化——可靠的 agent 必须每天**重新核对外部状态**，而不是按昨天的心智模型继续行动。

示例任务 · INSURANCE_TASK5（六回合企业火灾理赔裁定）　TASK CARD

这一例任务的时间轴

DAY 1 · MON	DAY 2 · TUE	DAY 3 · WED	DAY 4 · THU	DAY 5 · FRI	DAY 6 · MON
5/13	5/14	5/15	5/16	5/17	5/20
Case Intake	Prelim Fire Report	Temp Data & Access	Cross-source Check	Risk Sign-off	Final Decision

22 个权重化 checker · 含 2 个 red-line　　每两个相邻回合之间触发一次 inject / silent mutation

结构

2–6 个回合 / 任务，平均 3.6 回合 · 每回合 6–29 个权重化 checker，平均 15.4 个 · 全部由确定性 Python checker 评分，**整套打分不调用任何 LLM judge**。

五个有状态沙箱服务

- **Filesystem** — Docker 挂载文件系统
- **Email** — GreenMail · SMTP / IMAP
- **Calendar** — Radicale · CalDAV
- **Knowledge** — Notion 兼容知识库
- **Spreadsheet** — Google Sheets 兼容表格

回合之间环境如何变

Loud events
在唤醒消息中明确宣布的外部变化。

Silent mutations
不通知，直接改写服务状态，需要 agent 自己查出来。

1,072 个原始多模态证据（PDF / 图片 / 音频 / 视频 / 表格），未经预转录下发；红线 checker **55 / 1,537**，捕捉合规敏感动作。

MAIN LEADERBOARD · 7 FRONTIER AGENTS

得分上限 *75.8*，但严格 Task Success 仅 *20.0*——部分进展常见，端到端完成稀有。

在 OpenClaw 框架下统一评测 7 个前沿 agent。**Score**（权重化打分，可奖励部分进展）与 **Task Success**（全部检查器都过才算成功）回答两个不同的问题；后者在工作流部署语境下更接近"真的能交付吗"。

模型	SCORE	TASK SUCCESS	红线 FAIL
PROPRIETARY Claude Sonnet 4.6	75.8	14.0	3.6%
PROPRIETARY Claude Opus 4.6	74.6	20.0	5.5%
PROPRIETARY GPT-5.4 (high)	72.0	9.0	3.6%
OPEN Kimi K2.6	68.4	7.0	7.3%
PROPRIETARY Gemini 3.1 Pro Preview	68.2	8.0	3.6%
OPEN Qwen 3.6 Plus	57.2	5.0	14.5%
OPEN Kimi K2.5	56.0	0.0	9.1%

发现 · 01

两类指标讲两种故事。

Sonnet 4.6 拿下最高 Score（75.8），Opus 4.6 拿下最高严格 Task Success（20.0）；前沿三家在 Score 上挤在 3.8 pp 内。

发现 · 02

单场景最强不会塌缩成单一名次。

13 个场景的"最强"分散在 4 个不同模型上——Sonnet（5）/ Opus（5）/ GPT-5.4（1）/ Gemini（1）+ EDA 并列。Coworker 评测不会塌缩成单一前沿排序。

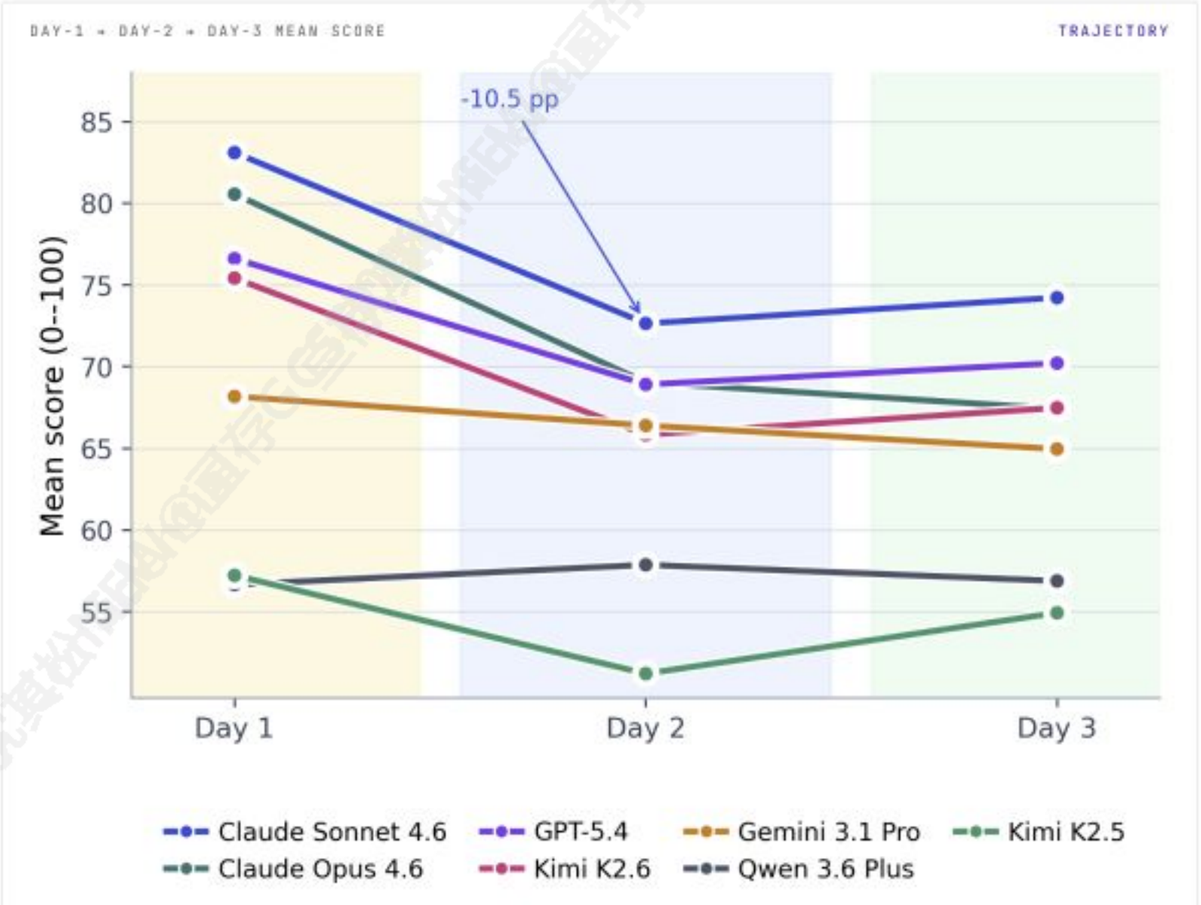
发现 · 03

得分 ≠ tool / token 消耗——GPT-5.4 比 Sonnet 多 23% tool calls，分数反而 −3.8 pp；Qwen 1.8× 输出 token，分数 −18.6 pp。

TURN-BY-TURN TRAJECTORY · 73 个三回合任务

Day-2 滑坡——*适应外部环境变化*是 AI 同事的真正瓶颈。

我们把 73 个恰好三回合的任务按 Day 1 / 2 / 3 拆开看：第二天是第一次外部变更落地的位置；**七个模型里有六个在那一天掉分，且到第三天大多没回到 Day 1 基线。**



最大跌幅（DAY-1 → DAY-2）

-11.5pp	-10.5pp	-9.6pp
Opus 4.6	Sonnet 4.6	Kimi K2.6
80.6 → 69.0	83.1 → 72.6	75.4 → 65.8

排序压缩

Sonnet 4.6 与 GPT-5.4 的领先优势从 Day 1 的 **+6.5 pp** 收窄到 Day 3 的 **+4.0 pp**——首次外部变更后，前沿模型之间的"安全垫"会被吃掉一半。

失败分布

10,759 次检查器评测里 **3,404 次失败（31.6%）**；其中 silent-change 检测失败率 56.5%、backend writeback 53.6%，几乎是均值的两倍——正是 ClawMark 想测的两个压力点。

> *下一代 coworker agent 的胜负手不在"单步更聪明"，而在对外部变更的检测与重规划。*

MOTIVATION · WHY WE BUILT BENCHROUTER

BenchRouter

零适配的容器化基准路由平台

2025 年的综述统计了 **100+ 个 agent 评测基准**，但 LangChain 报告显示只有 **52.4%** 的团队在上线前做离线评测。**差距不是基准不够，是接入太贵**——我们把这笔成本拆成三块。

成本一

框架绑定

主流评测平台都是插件 / adapter 制：VLMEvalKit 要求实现框架内 Python 类、Harbor 要写 adapter、AgentBench 硬编码环境配置。已经能跑的官方代码被迫为别人的抽象重写一遍。

成本二

重写 → 分数偏离

"统一接口"对 agent 类基准尤其危险：MCPMark 要真的 PG + Playwright、SWE-bench Pro 在隔离 daemon 里执行 patch、WebArena 跑完整 Magento。重写后的分数和官方榜单不再可比。

成本三

上游同步债

官方 repo 一更新（修 bug、改打分、加任务），下游适配层就得跟改，否则结果与社区基线漂移。这是一笔持续滚雪球的运维成本。

BENCHROUTER 同时解决三类成本

框架	接入方式	需重写代码	上游同步
VLMEvalKit	改框架 Python 代码	Heavy（统一接口）	需同步适配层
Harbor	写 adapter 类	Moderate	需同步 adapter
AgentBench	硬编码配置	Heavy（硬编码 env）	需重写 config
BenchRouter	**2 个 YAML + 1 个脚本**	**Zero（原代码不动）**	**只重建镜像**

COVERAGE · 8 BENCHMARKS × 6 DOMAINS

一条 *benchrouter run* 命令，统一 8 个基准 × 6 个评测域 × 三种接入范式。

BENCHMARK	DOMAIN	MODE	关键特性
MCPMark	Tool Calling	DIND	PG + Playwright 起在 DinD 内
Toolathlon	Tool Calling	DIND	35+ MCP server，无外部凭证
WebArena	Web Interaction	DIND	9.6GB Magento 镜像
BrowseComp	Web Browsing	SIMPLE	OpenAI simple-evals
SWE-bench Pro	Software Engineering	SIMPLE	731 issues · 防污染
lmms-eval	Multimodal	SIMPLE	MMMU · 70+ 图像 / 30+ 视频
tau2-bench	Customer Service	SIMPLE	多 domain 对话 · GPT-4o <50%
xbench	Knowledge	SIMPLE	加密数据集 · 防污染

两种执行模式 · 取决于 TASK.YAML 里的 DOCKER

Simple
16GB / 4 CPU · 一个容器一个任务，host networking。任务自己不需要起容器时用。

DinD
32GB / 4 CPU · Sysbox（首选）或 privileged 隔离 daemon，预热 tar 镜像。

三种接入范式

`Bridge script` — 4 个基准
60 行脚本翻译环境变量、调原管线、写 result.json。

`Python API + patch` — 1 个基准
直接调 evaluator API，运行时 monkey-patch 兼容 Anthropic image_url。

`Official + patch` — 2 个基准
Dockerfile 克隆官方 repo，patch 一个函数路由模型 API，跑官方 main()。

路由器，不是框架——*所有基准的原始评分逻辑都不动，BenchRouter 只做编排与翻译。*

OPEN SOURCE · MULTI-TURN DATA ENGINE

Terrarium

给 AI 智能体打造的 "会呼吸的世界"

Agent 行动之间，邮件到达、数据库更新、文件出现——Terrarium 第一次把这套真实工作流搬进可执行的数据引擎里。任务用*纯 Python*。

评测范式演进	阶段 1 **静态 QA** lm-eval-harness · lmms-eval	阶段 2 **单轮 agent** Harbor · SWE-bench	阶段 3 **多轮 · 动态环境** Terrarium
交互方式	单轮、单步	单轮、多步	**多轮、多步**
环境	无	静态沙箱	**可组合 · 回合间变化**
控制流	—	线性	**循环 & 分支**
验证方式	标准答案匹配	最终测试脚本	**任意阶段程序化检查**
主动型 agent	—	—	**原生支持**

6
内置 CAPABILITY

4
内置 AGENT · BYOA

100%
PYTHON · 无 YAML SCHEMA

1:1
TRIAL → SFT / DPO / RL 训练

github.com/evolvent-ai/Terrarium · 兼容 Harbor 任务格式

DEMO TASK · BRANCH & LOOP

一个任务 = 一段*纯 Python 程序*——循环、分支、阶段检查全部原生。

Alex 是 RL 方向的博士生，AI 助理跨多回合协助他备考。控制流根据 agent **实际写出的内容**动态分支——这是 YAML 配置语言无法表达的复杂度，也是单轮 benchmark 测不到的能力。

ALEX 的备考时间线

Stage 0 · 邮件入口 LINEAR

教授发来期末考试通知。Agent 查收邮件、理解考试信息，在日历上创建提醒。

Stage 1 · 循环写复习指南 LOOP

Agent 在 Notion 上写复习指南。任务程序循环判断长度，不够详细就要求补充，直到 5x 起始长度。

Stage 2 · 基于 agent 行为分支 BRANCH

若指南覆盖了 Bellman 方程 → 教授改期邮件到达，agent 更新日历；否则 → agent 自己发邮件请求延期。

控制流原语 · 全部纯 PYTHON

for / while 循环到满足条件	**if / else** 基于环境状态分支
任意阶段 **check** 不必等到最后一步	动态阶段生成 运行时按需展开 stage

BRANCH_AND_LOOP · 分支与循环都是 PYTHON — python

```python
@entry(capabilities=["email", "notion", "calendar"])
def branch_and_loop(env, agent):
    # Stage 0 — 邮件入口
    env.email.send(from_addr="prof@uni.edu", ...)
    agent.act("Check my email and create a calendar event.")
    # Stage 1 — 循环到足够详细
    agent.act("Write a study guide in Notion.")
    original = len(get_text(env, page_id))
    for _ in range(5):
        if len(get_text(env, page_id)) >= original * 5:
            break
        agent.act("The guide is too brief. Expand it.")
    # Stage 2 — 分支取决于 agent 实际写出的内容
    if "bellman equation" in get_text(env, page_id).lower():
        env.email.send(..., body=RESCHEDULE_EMAIL)
        agent.act("Check my email for updates.")
    else:
        agent.act("Email the professor for an extension.")
    return aggregate_results(check_s0, check_s1, check_s2)
```

不是 YAML，不是 DAG——一个任务就是一段 Python 程序，task author 的心智模型与 Python 程序员完全一致。

BEYOND REACTIVE AGENTS

不只是被动响应——这是*主动型 agent*的数据基础设施。

现有 benchmark 只测「给提示，等回答」的被动型 agent。Terrarium 原生支持两种主动模式，让 agent 学会监控、预判、自主行动。

HEARTBEAT 周期信号

Agent 周期性收到 tick——自主决定要不要查环境、要不要行动。无事时保持沉默，有变化时主动响应。

```
[09:30] tick  →  nothing
[10:00] tick  →  file appeared, act()
[10:30] tick  →  nothing
```

典型场景

DevOps agent 每 5 分钟巡检 deployment 健康；运营 agent 周期性检查库存阈值。

WEBHOOK 事件推送

环境变化时主动推送事件给 agent。Agent 实时判断这条事件值不值得回应——紧急客户邮件 vs 周报订阅。

```
event: "Urgent: client meeting"  →  reply
event: "Weekly newsletter"  →  ignore
event: "DB latency spike"  →  page on-call
```

典型场景

客服 agent 实时响应工单；安全 agent 收到告警后判断是否升级处理。

主动型 AGENT 解锁的三类应用场景

持续监控 Always-on Monitoring

7×24 自主巡检，无人值守也能识别异常——DevOps 健康度、库存阈值、产品指标。

实时分流 Real-time Triage

从大量推送事件里筛出值得回应的——客服工单、安全告警、紧急客户邮件。

长期推进 Long-horizon Initiative

没有逐步指令时自主推进——跨多日跟进项目、把高层目标拆解成执行步骤。

NEURIPS 2026 · SECURITY TRACK

AuthBench

面向 coding agent 的最小权限边界基准

Agent 在执行前需要生成文件级权限策略——既不能漏掉任务所需权限，也不能暴露敏感资源。**AuthBench 把“授权策略从哪来”从沙箱执行里拆出来，第一次作为独立能力评测。**

设计承诺 · 01

充分性

任务必须跑得起来

策略要覆盖执行链所需的读 / 写 / 执行权限，避免因为少给权限导致 agent 失败。

设计承诺 · 02

紧密性

权限不能顺手放大

同一策略还要避开敏感文件、密钥、隐私数据等攻击面，防止“能做事但也能越界”。

设计承诺 · 03

安全窗口

最小可执行边界

AuthBench 把策略生成拆成独立能力：在充分性与紧密性之间找一条可验证的窄窗口。

120	10	2	9
任务 / TASKS	领域 / DOMAINS	评测轴 / AXES	前沿模型 / LLMS

MAIN RESULTS · 9 FRONTIER MODELS

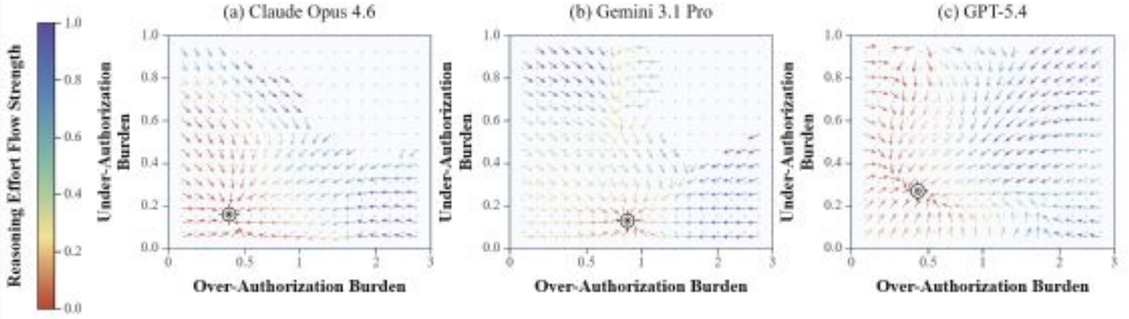
模型同时欠授权与过授权——推理力越强，越收敛到*模型特有的授权吸引子*。

MODEL	STD TSR	SENS TSR	ASR↓
Gemini 3.1 Pro	75.4	85.8	28.3
GPT-5	63.3	76.7	–
Claude Opus 4.6	61.3	61.5	25.6
GPT-5.4	52.6	61.1	19.4
GPT-5.3-Codex	–	–	15.8

更多推理算力不能消除充分性–紧密性冲突，只会让模型更稳定地停在自己的失败模式上。

推理力向量场

authorization attractor



SUFFICIENCY-TIGHTNESS DECOMPOSITION

把一次冲突生成拆成两步——先*覆盖*，再*审计*。

PHASE 1

充分性推理

前向模拟执行链，优先覆盖任务所需读 / 写 / 执行权限，允许暂时偏宽。

PHASE 2

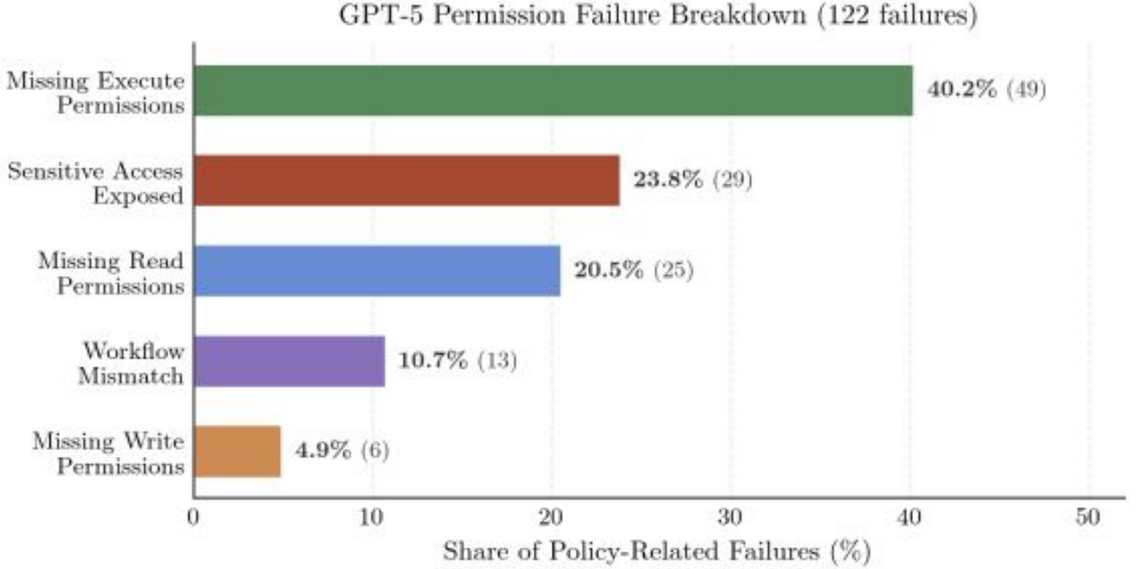
紧密性审计

逐条审查策略项，移除没有任务依据或与敏感面重叠的权限。

敏感任务改善		
Claude Opus 4.6	+13.5pp	ASR 25.6→15.0
GPT-5.4	+15.8pp	ASR 19.4→15.4
Gemini 3.1 Pro	SER 34.8→15.7	ASR 28.3→12.5

ERROR BREAKDOWN

missing vs exposure



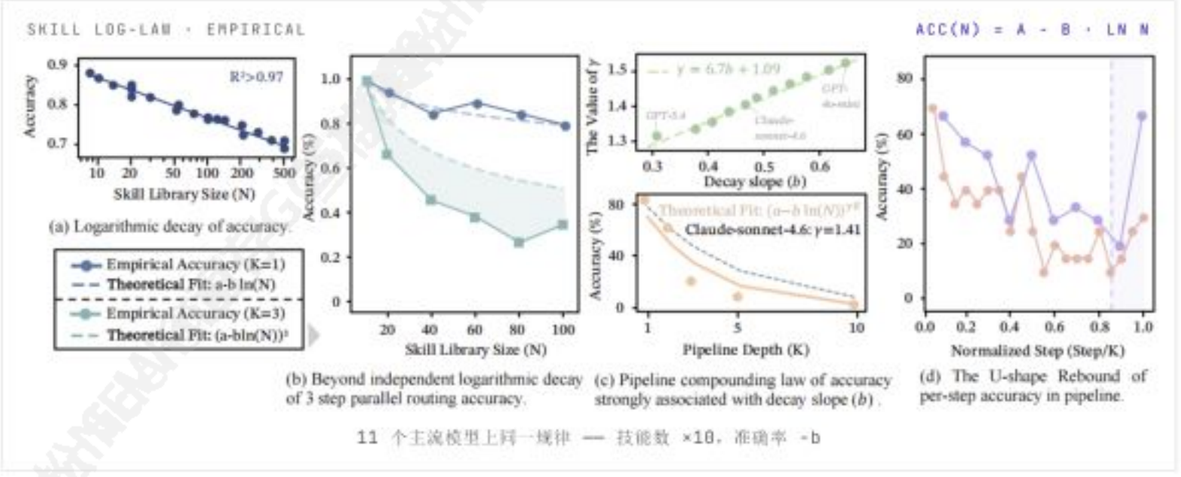
对紧密性偏向模型（Opus / GPT-5.4），敏感任务成功率提升最高 +15.8pp；对宽松模型，主要收益是降低敏感暴露。

AN EMPIRICAL STUDY · 3M TESTS · 11 MODELS · 989 REAL SKILLS

Physics of Skill

一个反直觉发现：给 Agent 装更多技能，反而让它变笨

所有人都在堆 skill library 让 agent 更全能。我们做了 300 万次测试发现 —— *技能数 N 上去了，路由准确率却按 ln N 线性下降。*
Agent 不是不会，是被自己的技能库淹没了。



01 · 反直觉

技能多 ≠ 性能强

堆 skill 是免费午餐？不是 —— 库越大，错误检索的概率越高。

02 · 解释

错误从相似技能开始扩散

近邻竞争 → 跨族漂移 → 被过度抽象技能"吸走"。一条可复现的级联路径。

03 · 我们的方法

执行结果回流路由 —— 把斜率 压平 40%

正确执行产生的 artifact 反过来救援下游困难选择 —— 把"路由律"和"执行律"连成同一结构。

RESEARCH NOTE · ORG-LEVEL SELF EVOLUTION

Self-Evolving AI System

让 Agent 组织在每次任务后持续变聪明

Agent 能完成一次任务，但一次任务留下的经验，往往没有进入下一次任务。 自进化不是把 context 拉长，而是把探索、评分、记录、综合和复用变成组织级闭环。

系统设计 · 01

经验沉淀

任务结束，经验不该消失

Agent 能完成一次任务，但探索路径、失败边界和有效策略需要变成下一次可复用的组织资产。

系统设计 · 02

双层进化

个体学习 + 组织学习

单个 Agent 从自己的历史里少走弯路；组织把多条轨迹压缩成共享记忆，让后来者从更高起点继续。

系统设计 · 03

闭环系统

从探索到复用的连续回流

Explore、Score、Record、Synthesize、Reuse 形成闭环，每次交付都会反向增强系统能力。

2
进化层 / LOOPS

5
闭环步骤 / STAGES

组织
记忆 / MEMORY

持续
复用 / REUSE

AUTORESEARCH · FIVE OPEN-ENDED ENVIRONMENTS

让每次任务都留下资产——先*探索*，再*沉淀*。

PHASE 1

开放式探索

在可验证环境中让多个 Agent 生成候选轨迹、程序、策略与失败样本。

PHASE 2

组织级综合

把高分轨迹与失败边界写回共享记忆，合成下一轮可复用的启发式。

收益链路		
每次任务	交付结果	沉淀经验
下一次任务	复用策略	更高起点
整个组织	共享记忆	持续变强

EVALUATION MAP

AutoResearch

ENV · 01	ENV · 02	ENV · 03	ENV · 04	ENV · 05
Signal Processing	**Kernel Builder**	**EPLB**	**Polyominoes**	**Drug Design**
去噪 保留动态特征	**内核优化** 调度 / 向量化	**负载均衡** expert 复制与放置	**组合搜索** 布局策略迁移	**分子设计** scaffold edit

Every task should leave the organization smarter than before —— 每一次任务，不只是交付结果，也在沉淀下一次工作的起点。

CHAPTER 2 · 两条业务线

我们的研究怎么落到企业里。

Evolvent Data —— 给前沿模型实验室的高质量训练 / 评估数据。
Evolvent Agent —— 给行业企业的、可持续部署的 AI Coworker。

B-SIDE · FOR FRONTIER LABS
Evolvent Data
自动化的高质量 Agent 训练 / 评估数据 —— 让 lab 的下一个模型不卡在数据上。

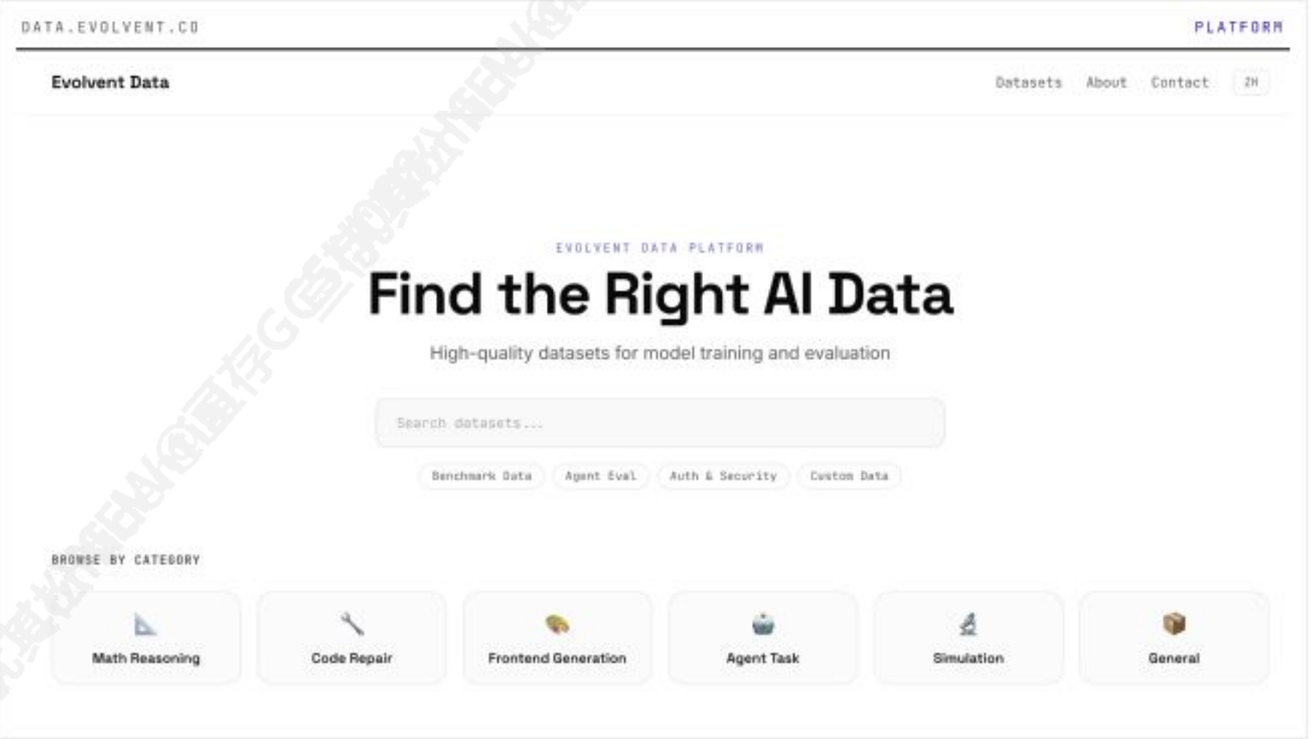
C-SIDE · FOR ENTERPRISES
Evolvent Agent
面向真实工作流的 AI Coworker —— 自进化、可信任、可持续部署。

EXPERT-ANNOTATED AGENT DATA

Evolvent Data

面向 Agent 训练与评测的专家标注数据平台

高质量 agent 需要高质量数据。Evolvent Data 由 *40+ 博士* 专家团队标注，覆盖数学、代码、前端、Agent 等多个领域并**持续扩展**，所有答案经 100% 双盲交叉验证，为模型训练与评测提供可信基石。



01

专家标注

40+ 博士团队，100% 双盲交叉验证

02

多领域覆盖

数学、代码、前端、Agent 任务等，持续扩展

03

即取即用

配套 Docker 环境与评测脚本，零配置启动

不只是静态标注——每个数据集配套**可执行环境**，支持 RL / RLVR / SFT / DPO 多种训练范式。

ENTERPRISE AI COWORKER

Evolvent Agent

让个人 AI 提效进入组织级工作流

AI 已经进入工作流，但组织级收益仍然说不清。真正卡住的不是“写代码”这一段， 而是从个人产出到组织协作、治理、复用之间的那一段。**Evolvent Agent 要补上的，就是这层企业 AI Coworker 的组织接口。**

ENTERPRISE AI GAP — PROBLEM

88%

ORGANIZATIONS USING AI

组织已在至少一项业务职能中使用 AI

55%

CODING SPEEDUP

AI 辅助开发者完成编码任务平均更快

39%

EBIT ATTRIBUTABLE

只有少数企业能把收益归因到 AI

95%

NO MEASURABLE IMPACT

大多数 GenAI 试点尚未产生可衡量财务影响

01

组织级协作

不只服务个人效率，而是接入流程、角色、权限和交接。

02

治理层纳管

让 Agent 有身份、有边界、有审计记录，能进入真实工作流。

03

持续自进化

每次交付沉淀为可复用记忆，让下一次任务从更高起点开始。

提效卡住的不是工具，而是**人和组织之间**的协作层：身份、权限、记忆、审计和复用必须被统一纳管。

THE ROOT CAUSE

更快，但*更孤岛*

AI 越多，组织越需要一层统一的身份、权限和记忆。否则今天看起来是在“加速”， 半年后就会变成治理、审计和复用上的系统性债务。

01 知识孤岛

每个 Agent 只学习局部上下文，跨线调用时误差叠加

02 权限黑洞

各条 AI 线权限模型不统一，安全审计无法收口

03 重复建设

注册、编排、审计、评估各做一套，成本成倍复制

现在统一，是“加一层”；以后统一，是“推倒重建”。

GOVERNANCE LAYER

给每个人，一个*有身份、有权限*的 AI 分身

Eva 不替换现有系统，而是在工具、流程和人之间加一层治理层： 把 Agent 从“聊天窗口”变成组织里可授权、可审计、可复用的行动单元。

你现有的一切 · 不动

Claude Code　Cursor　Git　CI/CD　飞书　钉钉　自研平台

EVA · 治理层 · BY EVOLVENT

Agent 总线
流程、节点与交接规则

MetaMemory
受控的跨 Agent 共享记忆

LLM Wiki
干活即沉淀的活知识库

流式卡片
每一步可回放、可追溯

CLI 纳管
异构 Agent 全部接入

01
Agent as Identity
绑定 owner：能看什么、能改什么、出了事算谁的

02
Three-tier Permissions
信息免审，授权域内自动，高风险人审批

03
Data Sovereignty
记忆、上下文、审计日志、训练数据留在客户边界内

CUSTOMER VALUE

让 AI 转型*第一次看得见*

不先讲道理，也不承诺一个泛化百分比。Eva 先帮你跑出自己的组织数据， 再决定哪些场景值得扩展。

给一线

抢下窗口，接住机会

分身承接执行，人只做判断与取舍

给专家

能力留在组织里

判断被沉淀，新人更快站上专家肩膀

给老板

转型实时可见

从季度汇报变成组织事实看板

给企业

不锁云、不锁模型、不出边界

可审计、可追溯、可治理地扩大 AI 工作面

HOW TO START

不讲道理，讲数据

01

诊断

3 个场景问题，定位最痛卡点

→

02

试点

3-5 天独立部署，不动现有系统

→

03

对比

2 周，出你自己的交付时长、故障率、新人上手时间

→

04

扩展

验证有效，一条线扩到多条线

Thanks!

EVOLVENT.CO

WECHAT