苏菲拉底 THE SOPHIE LAB

本期追问

创始技术押错方向后，凭什么还能长成世界领导者?

决定公司命运的，是具体技术还是对世界的独特信念?

当技术每天都在变，CEO为什么必须亲自下到一线?

视频精读 VIDEO STUDY

Jensen Huang: The Mindset That Built NVIDIA

来源：YouTube 视频（https://www.youtube.com/watch?v=l4B37S1dyQQ）

节目发布：2026-07-26 · 全文 71 段 · 页边注释 68 条 · 生词词组 89 条

好视频，值得精读

目录 CONTENTS

导读 视频总结

一句话概括

黄仁勋在 Startup School 2026 访谈中回顾了 NVIDIA 从创始技术选错、靠 Sega 的 500 万美元续命，到凭借"加速算法领域"这一核心信念抓住深度学习浪潮的历程，并向创业者传达了"学习是最大的超能力、系统思维是未来最重要技能、how hard can it be"的心态哲学。

核心要点

1. **NVIDIA 创始时的技术选择完全错误**：1993 年创立时选择的 3D 图形算法根本行不通，1995 年才发现，且团队当时没人知道正确做法——黄仁勋去 Fry's 买了三本 OpenGL 教科书发给工程师，从教科书起步最终成为图形领域的世界领导者。
2. **公司真正的正确大方向是"加速计算"**：用加速器增强 CPU 去解决原本无法解决的问题——3D 图形只是第一个"算法领域"，之后是分子动力学、图像处理、深度学习等；伟大公司靠的是对世界的独特且深信的视角，而非某项具体技术。
3. **Sega 的 500 万美元救了公司**：黄仁勋坦承技术有缺陷、无法完成 Dreamcast 合同，却仍请求对方支付全款；Sega CEO 因为信任他的诚实而给了钱，这笔钱让 NVIDIA 活到了转机——"投资的是人，不是公司"。
4. **AlexNet 的真正意义不是 AlexNet 本身**：而是"通用函数逼近器"的发现——深度学习是一种全新的软件生产方式，会重塑从处理器、中间件到应用的整个"五层蛋糕"，NVIDIA 由此在约 15 年前就开始布局计算机视觉、机器人和自动驾驶。
5. **CEO 应该"把车适配给自己开"**：不必用传统管理方法，创始人是 F1 赛车手，组织应不断调整以适配自己；至于继任者，"下一任 CEO 自己会重塑公司"。深入一线（读论文、直接与科学家交流）是为了对快速变化的技术保持"冲浪者式"的第一性原理手感。
6. **AI 消灭的是任务，不是工作**：编码被自动化后软件工程师岗位反而年增 10%，放射科读片被 AI 接管后放射科岗位增长约 20%，律师助理岗位也在增长——因为积压的需求（想法、病人、案件）极大，生产力提升驱动增长，增长带来更多就业。
7. **智能体（Agent）最需要的突破是"可控性"**：能对计划文件改一个词、改一个像素/一个三角形/一条连线，就产生精确的增量差异，然后让 agent 重新生成其余部分；agent 不需要 100% 准确，80%~99% 再由人补足即可。
8. **物理 AI 的 "ChatGPT 时刻"几年前已经发生**：看到生成式 AI 能生成手指运动的视频，就意味着机器人关节控制近在眼前；路径是 real-to-sim（构建学习/评测环境）→ 物理仿真（Isaac Sim、Cosmos）→ sim-to-real（强化学习）。自动驾驶是第一个规模化落地场景，NVIDIA 的物理 AI 业务已近 100 亿美元，将成为下一个千亿美元业务。
9. **开源是行业根基**：没有 Linux、Kubernetes、PyTorch 就没有现代 AI；NVIDIA 支持 Open Claw、Hermes 等开源项目，主张每个公司都应能构建自己的领域专用 AI。
10. **给年轻人的建议**：简单的事（写代码）会被自动化，但硬科学（物理、化学、生物、计算机工程）和交叉领域的难题永远存在；系统思维是编排百万 agent 的关键能力；心态上要相信"how hard can it be"，让苦

难一点点来，每天只需战胜今天——韧性（resilience）是最重要的品质。

章节时间线

- **[0:07] 开场** — Startup School 2026，欢迎黄仁勋登台
- **[1:09] 创始技术完全选错** — 1993 年的 3D 图形算法是错的，1995 年靠买教科书重学，重新发明计算机图形学
- **[5:06] 公司的核心信念** — 不是造好芯片，而是加速算法领域；伟大公司源于独特且深信的世界观
- **[7:06] Sega 救命钱的故事** — 坦承合同无法履行却请求全款，Irimajiri 的 500 万美元让 NVIDIA 活了下来；1999 年上市估值 3 亿美元
- **[10:01] 看见 AlexNet 的本质** — 通用函数逼近器的发现，深度学习是新的软件方式，由此布局视觉、机器人、自动驾驶
- **[13:30] 创始人模式与组织哲学** — 好奇心驱动、深入一线、CEO 是服务者；"把 F1 赛车适配给自己开"，持续为自己重塑公司
- **[19:12] 前沿方向：系统思维与 agent 可控性** — 底层工作将被 agent 自动化，系统思维成为最有用技能；细粒度控制是 agent 最大的待解突破
- **[23:26] 为什么 NVIDIA 深入研究 agent** — agent 是新软件，决定未来 5~10 年的计算架构；内部让"千种工具齐放"加速公司；从思维链早期信号推演出低数据量的"会思考的自动驾驶"（Alpaca）
- **[27:54] 开源生态与"人人构建自己的 AI"** — OpenCL 是"Linux 时刻"，支持 Hermes/Open Claw，鼓励企业构建领域专用 AI
- **[30:31] AI 与就业经济** — AI 自动化的是任务不是工作；软件工程师、放射科、律师助理岗位都在增长，生产力→增长→更多就业
- **[34:00] 物理 AI 与机器人** — 生成视频即预示机器人关节控制；real-to-sim-to-real 三段路径；自动驾驶先落地，物理 AI 业务近百亿美元，十年内成大产业
- **[39:00] 首次发帖 X 与开源宣言** — 2026 年才发第一条帖子，因开源议题太重要而克服"内向"
- **[40:37] 年轻人该学什么** — 简单的会被自动化，硬科学与系统思维永不过时；"stay in school"
- **[43:51] 给 18~22 岁自己的电报** — 创业时的恐惧与无知都不重要，现在是 60 年来创业最好的时机；"how hard can it be"、每天只需战胜今天、韧性至上

结论与值得注意的细节

结论：这场访谈的主线是"学习能力压倒一切"——NVIDIA 的起点是一个错误的技术选择加三本教科书，而它能穿越 3D 图形、加速计算、深度学习到物理 AI 的每一次浪潮，靠的是直面现实、第一性原理推演（"if this, then what?"）和把未来 5~10 年当作当下来设计系统的习惯。对创业者的实操建议集中在两点：培养系统思维（因为底层执行将由 agent 完成），以及用"how hard can it be"的心态启动、用日复一日的韧性坚持。

值得注意的细节：

- NVIDIA 当年差点承接 Sega Dreamcast 的开发，是因技术缺陷主动退出的；那笔 500 万美元合同款是黄仁勋"既说做不了、又要全款"谈来的。

- 黄仁勋称"words are thoughts"（词语即思想），认同 markdown 文件等文本形式承载智能，agent 通过更新记忆文件已具备"粗粒度的递归自我改进"。
- NVIDIA 内部不指定统一编码工具，Claude Code、Codex、Cursor、Cognition 等"百花齐放"，公司从使用数据中学习。
- 他提到开源自动驾驶栈的原因很具体：农业、邮件配送、仓库 AMR 等市场单个都不够大，但足够多样，值得开放整个栈。
- 关于接班问题的回答颇为坦率："等我在任上去世的那天，他们为下一任 CEO 重塑公司就是了。"
- 类比很密集：CEO 像冲浪者（读浪、读风、时机），组织像为车手定制的 F1 赛车，长除法被自动化类比编码被自动化。

第一部分　全文对照 · 附页边注释与生词标注

01　开场：黄仁勋登台

0:07 Garry Tan

Welcome to Startup School 2026. Now, let's get started. Please join me in welcoming to the stage the founder and CEO of Nvidia, Jensen Huang. >> [music] [music]

欢迎来到 2026 年创业学校（Startup School）。现在，我们开始吧。请和我一起欢迎登台的 Nvidia 创始人兼 CEO，黄仁勋。>> [音乐][音乐]

Startup School 是创业孵化器 Y Combinator 面向早期创业者的公开活动。英伟达由黄仁勋与两位伙伴于 1993 年创立，现为全球市值最高的芯片公司。

02　创始技术全错与教科书自救

0:34 Garry Tan

>> Hey Jensen. >> [music] >> Please. Everybody. >> Oh my god. This is a surreal moment for me. Thank you. Thank you for being here, Jensen. >> I'm delighted to do it. It's great to be here. >> [cheering] >> Apparently, if you're here, you are going to make it. So, I'm happy I'm here. >> [laughter] >> Oh, Jensen. Well, for the students who only know Nvidia at this as a company at the center of AI, what part of the early Nvidia story do they most need to understand? >> The thing that most people don't believe is that the choice of our technology that we started the company with was absolutely wrong.

>> 嘿，Jensen。>> [音乐] >> 请。各位。>> 我的天。这一刻对我来说太不真实了。谢谢你。谢谢你能来，Jensen。>> 我很乐意来。能来这里真好。>> [欢呼] >> 看来，只要你人在这儿，你就一定能成。所以，我很高兴我来了。>> [笑声] >> 哦，Jensen。呃，对于那些只知道 Nvidia 如今是一家处在 AI 中心的公司的学生来说，呃，早期 Nvidia 的故事里，哪一部分是他们最需要理解的？ >> 大多数人不敢相信的一点是，呃，我们创办公司时所选择的那项技术，是彻头彻尾的错误。

全场第一个反直觉论点：英伟达赖以创立的技术选择『完全是错的』。这为后文『学习能力比初始技术更重要』的核心结论埋下伏笔。

surreal /səˈriəl/ *adj.*
超现实的；恍如梦境般不真实的

1:31 黄仁勋

And so, we had started with the idea that we would reinvent 3D graphics. Well, the company's philosophy and perspective was that the general purpose computers, the CPUs, were really useful, but if we could augment it with accelerators, we could solve problems that otherwise too hard to solve. And one of the first problems we chose was 3D graphics. And we And during that time 1993 the PC was just rumored to be coming. And our big idea was that we would turn every single personal computer into a game console because we grew up in the era of game consoles.

我们一开始的想法是要重新发明 3D 图形。嗯，公司的理念和视角是：通用计算机，也就是 CPU，确实非常有用，但如果我们能用加速器去增强它，我们就能解决那些原本难以解决的问题。而我们选择的最初的问题之一，就是 3D 图形。当时是 1993 年，PC 还只是传闻中即将到来的东西。而我们的宏大构想是，把每一台个人电脑都变成一台游戏主机，因为我们是在游戏主机的年代里长大的。

1993 年 PC 游戏市场尚未成型，3D 图形主要存在于 SGI 工作站与街机、主机上。『CPU 加速器』的异构计算理念后来演化为 GPU 与 CUDA，是英伟达三十年战略主线。

augment /ɔːgˈmɛnt/ *v.*
增强，扩充（在原有基础上加强）

accelerators /ækˈsɛləreɪtərz/ *n.*
加速器（辅助 CPU 的专用计算硬件）

2:12

And so we thought you know what if we could design a system that would fit into the personal computer and it would turn it into a game console. And so we thought we would reinvent the algorithm that would require these large supercomputers and we would fit it into the PC. And we came up with some new algorithms. And we were excited about it. We believed in it. It was we reasoned about it in a thoughtful way. And we went to start the company to go build it. Well, it turns out the algorithm was exactly wrong.

所以我们想，如果我们能设计一套系统，装进个人电脑里，把它变成一台游戏主机呢？于是我们想，我们要重新发明那种原本需要大型超级计算机才能跑的算法，把它塞进 PC 里。我们提出了一些新的算法。我们对此很兴奋。我们相信它。我们是经过深思熟虑地推演出来的。然后我们就去创办公司，把它做出来。结果呢，那个算法完全是错的。

注意论证顺序：深思熟虑地推理、坚信、创业——结果仍然错了。说明再认真也可能选错路线，重点在后续如何应对，而非避免犯错。

2:47

And the technology that founded the company turns out to be exactly wrong. And so in 1995 we realized that and it was almost too late because by then there were some 35 40 other companies that were building 3D graphics for PCs. And so we realized that it didn't work and I went back to the company and we were at the company I said what are we going to do? It doesn't work and we were all talking about it and I said look we won't have a company if we don't confront the fact that this doesn't work and start working towards the right algorithm.

那项作为公司立身之本的技术，结果证明是完全错误的。于是在 1995 年，呃，我们意识到了这一点，而那时候几乎已经太晚了，因为当时已经有大约 35 到 40 家其他公司当时在为 PC 做 3D 图形。于是我们意识到这条路走不通，我回到公司，我们几个人在公司里，我说，我们该怎么办？这行不通啊。大家都在讨论这件事，我说，你们看，呃，我们、我们，呃，如果我们不正视这条路走不通这个事实，并且开始朝着正确的算法方向努力，我们这家公司就活不下去了。

1995 年前后 PC 3D 图形赛道有三四十家公司（3dfx、S3、ATI 等），最终几乎全部消亡，只有英伟达与 ATI（后被 AMD 收购）存活。

confront /kənˈfrʌnt/ *v.*
直面，正视（confront the fact/reality 直面事实）

3:25

And then somebody told me it turns out none of us knew how to do it the right way. And not only did we choose the wrong technology we didn't know how to do it the right way. And so that was a big day for me. I had a couple of couple of $60 you know a couple of $100 in my pocket and so I went down to Fry's and I bought three textbooks. And the textbooks was about OpenGL and how to design pipelines. I brought it back to the company and gave it to the engineers, and here we are. we reinvented computer graphics, we're the world leader in modern computer graphics, we invented most of the major breakthroughs in the last 25 years.

然后有人告诉我，结果我们谁都不知道怎么用正确的方式去做。我们不仅选错了技术，还压根不知道正确的做法是什么。所以那对我来说是很重要的一天。我口袋里揣着几十美元，你懂的，也就一两百美元，然后我跑去 Fry's，买了三本教科书。还有教科书讲的是 OpenGL，以及怎么设计渲染管线。我把它带回公司，交给工程师们，然后就有了今天。呃，我们重新发明了计算机图形学，我们是现代计算机图形学的世界领导者，我们发明了过去 25 年里大部分重大突破。

Fry's 是硅谷标志性电子连锁店（2021 年停业）；OpenGL 是 SGI 主导的 3D 图形行业标准接口。『图形巨头的起点是三本教科书』是英伟达历史上最著名的细节之一。

pipelines /ˈpaɪplaɪnz/ *n.*
管线；（图形学）渲染流水线

4:10

Everybody would have thought that Nvidia is you know started out as world leaders in 3D graphics, and we learned it from a textbook. And so we actually started the company, raised money, and bought textbooks. When you think about it. And so the big lesson is that for me is technology is changing all the time, and so long as you're able to confront the reality, so long as you are able to learn, the technology itself actually doesn't matter. And so since then Nvidia has been, you know, inventing all kinds of technology since, all kinds of technology we've never really done before, and we approach everything with the same attitude, you know, this is if it's important to do, we're going to go learn it, and how hard can it be?

所有人都会以为，英伟达一开始就是 3D 图形领域的世界领导者，而其实我们是从一本教科书里学来的。教科书。所以我们其实是先创办了公司、融了资，然后去买教科书。你仔细想想。所以对我来说，最大的教训是：技术一直在变，只要你能够直面现实，只要你愿意去学，技术本身其实并不重要。所以从那以后，英伟达一直在发明各种各样的技术，很多都是我们以前从来没做过的，而我们对待每件事的态度都一样，那就是，如果这件事值得做，我们就去学，能有多难呢?

核心命题：技术不断更替，公司的根基不是某项技术，而是『直面现实加快速学习』的能力。『能有多难呢』这句口号在结尾会再次出现，贯穿全场。

03 大理念：加速算法领域

4:57 Garry Tan

And it always turns out to be much harder than we expect. but you go into it with the attitude, how hard can it be? >> I mean, backstage we were talking about how I mean, we were talking with some of the top YC companies, and you were saying that each one has an expertise in like a domain that you have an you and Nvidia have an expertise in, and they're all just I forget what you said, it was like an algorithmic domain of a sort. And so it sounds like 3D graphics was merely the first of an algorithmic domain.

结果呢，总是比我们预想的要难得多。但你一开始就抱着这样的心态：能有多难呢? >> 我的意思是，我们在后台聊到，我们跟一些顶尖的 YC 公司交流时，你说他们每一家都在某个领域有专长，而你和英伟达在这个领域也有专长，它们都只是——我忘了你刚才怎么说的，好像是某种算法领域。所以听起来 3D图形只不过是众多算法领域中的第一个。

YC 即 Y Combinator，最著名的创业孵化器，曾孵化 Airbnb、Stripe、Dropbox。主持人把话题从 3D 图形引向『算法领域』这一更一般的概念。

5:29 黄仁勋

>> That's right. >> came from a textbook, but then, you know, anyone could have read that textbook. You created >> Particle physics, fluid dynamics, yeah. >> But you created the thing that people want, like the end product that people want to pay a lot of money for. >> The big idea of the company that was spot-on is that it is possible to augment the CPU to solve problems that otherwise are too difficult to solve. And molecular dynamics is one of them, image processing is one of them, inverse physics is another one. And so all kinds of different algorithms, of course deep learning is one of the major ones. And in order to create the company that we have today, we realized early on that it's not about building a great chip, it's about accelerating an algorithm domain.

>> 没错。>> 是来自一本教科书，但你知道，谁都可以读那本教科书。是你创造了 >> 粒子物理、流体动力学，对。>> 但是你创造了人们想要的东西，就是那种大家愿意花大价钱买的最终产品。因为。>> 这家公司最了不起、也完全押对的想法是：可以对 CPU 进行增强，去解决那些否则就太难解决了。分子动力学是其中之一，图像处理也是其中之一，逆物理是另一个。所以有各种各样的算法，当然深度学习是其中最重要的一个。为了打造出我们今天这家公司，我们很早就意识到，关键不在于做出一颗很棒的芯片，而在于加速一个算法领域。

理解英伟达战略的关键句：卖点不是芯片本身，而是『加速一个算法领域』——围绕分子动力学、图像处理、深度学习等领域构建软硬件全栈（即 CUDA 生态）。

spot-on /ˌspɑːt ˈɑːn/ *adj.*
完全正确的，精准的（口语）

molecular dynamics *phr.*
分子动力学（模拟分子运动的计算方法）

6:17

And so one of the things I've always believed in is what makes great companies is a unique perspective about the world that you deeply believe in. It's not so much the technology, it's not so much the market even. those things all matter and if you have the right technology for the right market at the right time, your life is going to be a lot easier. A high-level vision about the future of some important thing, a perspective about it that's somehow unique, that you deeply believe in and ideally pursuing that vision is hard to do, those are kind of good combinations. In our case, we realized that accelerated computing was going to be important. And accelerated computing turns out to be very important and our realization is everything to do with algorithm, not the chip. turns out to be exactly right.

所以我一直坚信的一点是，伟大的公司源于你对这个世界有一个独特的、并且你深信不疑的视角。这不完全是技术的问题，甚至也不完全是市场的问题。这些当然都重要，如果你在对的时间、为对的市场、拿出对的技术，那你的日子会好过得多。对某件重要事物的未来有一个高层次的愿景，一个多少有些独特的视角，一个你深信不疑、而且理想情况下追求起来还很难实现的愿景，这些是很好的组合。就我们而言，我们意识到加速计算会变得很重要。而加速计算事实证明确实非常重要，而且我们的那个洞见——一切都关乎算法，而不是芯片——事实证明完全正确。

创业方法论：伟大公司等于『对世界的独特视角＋深信不疑＋愿景难以实现』。『难做』反而是优点，因为它构成天然壁垒，挡住跟风者。

04 世嘉危机与救命五百万

7:06 Garry Tan

>> So you've said a lot about I guess the hardships of a founder. are there a few stories that really jump out at you? I mean there the people in this room would love to start a company, but you know, are they really prepared for eating glass and you know, possibly having to shut down the company, like things going wrong? Like what are some of the pivotal moments that really jump out at you? I think you were just in Japan, right? And >> Yeah. >> you were sort of honoring Sega, was it? So I feel like that was a really powerful story.

>> 你讲了很多关于创业者的艰难。有没有哪几个故事特别让你难忘？我是说，在座的很多人都想创业，但你知道，他们真的准备好了要「吃玻璃」，甚至可能得关掉公司、事情全乱套吗？有哪些关键时刻让你特别难忘？我记得你刚去过日本，对吧？然后…… >> 对。>> 你当时算是在向世嘉（Sega）致敬，是吧？我觉得那是个很有力量的故事。

『吞玻璃』（eating glass）出自马斯克名言『创业就像嚼着玻璃凝视深渊』，指创业的极端痛苦。主持人在为下面的世嘉故事做铺垫。

pivotal /ˈpɪvətl/ *adj.*
关键性的，起决定作用的

eating glass *phr.*
吞玻璃（喻创业的极端艰辛，源自马斯克名言）

7:38 黄仁勋

>> The project that led us to realize the algorithm we chose was wrong was a partnership with Sega. Sega had contracted us to build the game console after Saturn that turned out to have been Dreamcast. I don't know if Does anybody know what Dreamcast is? Okay. So, we did not build Dreamcast. We were originally supposed to build Dreamcast. But because our algorithm and our technology was fundamentally flawed, I went to Japan and I told Irimajiri-san, the CEO at the time, that the contract that they gave us was a like $12 million contract, we will not be able to fulfill because the technology doesn't work.

>> 让我们意识到自己选错算法的那个项目，就是和世嘉的合作。世嘉当时委托我们做土星（Saturn）之后的那台游戏主机，后来那台机器就是 Dreamcast。不知道有没有人知道 Dreamcast 是什么？好的。所以，我们没有做出 Dreamcast。本来是该由我们来做 Dreamcast 的。但因为我们的算法和技术存在根本性缺陷，我去了日本，告诉入交先生——当时的 CEO——他们给我们的那份合同，大概是一份 1200 万美元的合同，我们没办法履约，因为这项技术根本行不通。

世嘉土星（1994）之后的主机即 Dreamcast（1998）。英伟达因技术路线错误主动放弃这份合同。入交昭一郎曾是本田传奇工程师，时任世嘉社长。

flawed /flɔːd/ *adj.*
有缺陷的（fundamentally flawed 存在根本缺陷）

fulfill /fʊlˈfɪl/ *v.*
履行（合同、承诺）；实现

8:26

And I told him the reasons why. And then I advised that they choose somebody else to do it. but then I asked them I told him that I unfortunately still need the money. And he asked me, "Hey, you know, the conver- You could just imagine the conversation. So, what you're telling me is what I contracted you to do, you can't do, but you would like all the money on the contract." And I said, "You got it. That's exactly right." >> [laughter] >> But obviously I was polite. I was I was humble.

我把原因都告诉了他。然后我建议他们另找别人来做。但接着我又跟他说，很不幸，我仍然需要那笔钱。他就问我，「嘿，你知道那个对话……你完全可以想象那个场面。所以你的意思是，我委托你做的事，你做不了，但你还是想要合同上的全部款项。」我说：「没错，就是这样。」>> [笑声] >> 但我当然是很有礼貌的，我态度很谦卑。

谈判的荒诞之处由黄仁勋自己点破：『活儿干不了，钱还想全要。』这段自嘲式叙述对应英伟达历史上真实的生死关头。

8:57

And he realized that I was honest and everything made sense. I And if he didn't give us the money, we'd be out of business. And I think that this happens in this room. You don't invest in companies, you invest in people. And what Irimajiri recognized was here's, you know, somebody and a company that he trusted in the first place the contract and that he believed in and that he would love to see, you know, make it make it to the next day. And so, that $5 million kept us alive and, you know, gave me enough time to discover what to do.

他也意识到我是诚实的，而且一切都说得通。如果他不给我们那笔钱，我们就得倒闭。我觉得这种事在座各位身上也会发生。你投的不是公司，你投的是人。入交当时看到的是：这是一个他一开始就信任、才把合同给他的人和公司，是他所相信的，是他希望能撑下去、活到明天的公司。所以那 500 万美元让我们活了下来，也给了我足够的时间去搞清楚该怎么做。

『你们投的不是公司，是人』——直接点题台下的投资逻辑。世嘉最终支付约 500 万美元，让英伟达撑到找到正确路线（后来的 Riva 系列芯片）。

9:37 Garry Tan

>> And then I guess if they held they sold it for 15 million I heard. >> Yeah, they sold it the moment we went public. when Nvidia went public our valuation was 300 million dollars. 300 million dollars in 1999. That was real money. >> I think it's north of a trillion dollars now or so. >> It's more than a trillion, yeah. >> Yeah, that's wild. So, you're sort of the core, you know, I we like to say that you're the man who controls the spice. you know, before that, you know, I don't think anyone could have really predicted per se how important GPUs and you know, the technology you built would be for this AI revolution.

>> 然后我听说，他们当时把那部分股份以 1500 万卖掉了。>> 对，我们一上市他们就卖了。英伟达上市时，我们的估值是 3 亿美元。1999 年的 3 亿美元，那可是一笔真金白银。>> 我觉得现在应该是超过一万亿美元了吧。>> 是超过一万亿了，对。>> 是啊，太夸张了。所以你算是处在核心位置，我们喜欢说，你是那个「掌控香料的人」。在那之前，我觉得没人真的能预料到 GPU、以及你们做出来的这些技术，对这场 AI 革命会有多重要。

1999 年英伟达以 3 亿美元估值上市，如今市值超万亿。『掌控香料的人』出自科幻小说《沙丘》：谁控制香料谁就控制宇宙——暗指 GPU 是 AI 时代的战略资源。

valuation /ˌvæljuˈeɪʃən/ *n.*
估值

north of *phr.*
超过，多于（口语，用于数字前）

per se /ˌpɜr ˈseɪ/ *adv.*
本身，就其本身而言（拉丁语借词）

05 AlexNet：通用函数逼近器

10:21 黄仁勋

You know, what did you see to I mean, was it the accelerator and being in the right place right time or surely there were a lot of things that led up to that allowed you to sort of capture this position? >> Yeah. I saw AlexNet just like everybody else saw AlexNet. And I but remember, our lens of the world, my view of the world was always looking for algorithms. And that algorithm the algorithm could be NAMDI, the algorithm could be VASP, the algorithm could be OpenGL. You know, it could be SQL, some domain specific language, some algorithm.

那你当时看到了什么？我是说，是因为做加速器、又恰好在对的时间站在对的位置，还是说其实有很多事情层层铺垫，才让你们占据了今天这个位置？>> 是的。我和其他所有人一样看到了 AlexNet。但要记住，我们看世界的视角、我看世界的方式，一直都是在寻找算法。那个算法可以是 NAMD，可以是 VASP，也可以是 OpenGL。你知道，也可以是 SQL、某种领域专用语言、某种算法。

AlexNet：2012 年 Hinton 团队用两块英伟达消费级 GPU 训练的深度卷积网络，在 ImageNet 竞赛大幅夺冠，引爆深度学习革命。NAMD、VASP 是科学计算领域的知名软件。

lens /lɛnz/ *n.*
透镜；（喻）看问题的视角（our lens of the world）

11:00

And so my lens of the world was always looking for some problem that we might be able to help solve. So, when AlexNet came along, the algorithm was deep learning. And so, the question is what is this algorithm and why does it matter? Why was it so effective and what else can it do? And if you were to scale algorithms and scale it beyond that, what could it solve that otherwise you can't solve today? And the breakthrough for us was realizing that AlexNet was not AlexNet. That AlexNet was an approach with deep learning that allows you to learn any function. And so, 15 years ago, I was telling everybody that, "Hey, guess what? We just learned the universal function approximator."

所以我看世界的视角，一直是在寻找某个我们也许能帮上忙去解决的问题。所以当AlexNet 出现时，那个算法就是深度学习。于是问题就变成：这是个什么算法？它为什么重要？为什么它这么有效？它还能做什么？如果你把算法规模化，甚至推得更远，它能解决哪些今天解决不了的问题？对我们来说，真正的突破在于意识到 AlexNet 不只是 AlexNet。AlexNet 是一种用深度学习去学习任意函数的方法。所以 15 年前，我就跟所有人说：「嘿，你猜怎么着？我们刚刚学会了通用函数逼近器。」

关键的抽象跳跃：别人看到『一个识图模型』，黄仁勋看到『通用函数逼近器』——神经网络理论上可拟合任意函数。视角高一层，结论就完全不同。

universal function approximator *phr.*
通用函数逼近器（神经网络可拟合任意函数的理论性质）

11:50

We just discovered the universal function approximator. We can give it We could, you know, give it the answer for almost any function and it could learn what the function is. And for a lot of functions, you don't have to be precise. And in fact, it's impossible to be precise. And so, most of the interesting problems are imprecise in this way. And so, the day that we realized we have a universal function approximator, the question then is what is that what is that what is that do to the computing stack? What does that happen to software? What are the industries that this could impact? So on and so forth.

我们刚刚发现了通用函数逼近器。我们可以把几乎任何函数的答案喂给它，它就能学出这个函数是什么。而且对很多函数来说，你并不需要精确。事实上，精确是不可能的。所以，大多数有意思的问题，本质上就是这种不精确的问题。所以，在我们意识到自己拥有了一个通用函数逼近器的那一天，接下来的问题就是：这对整个计算堆栈意味着什么？软件会变成什么样？会有哪些行业因此受到影响？等等等等。

> 反直觉观点：大多数有价值的问题（视觉、语言、驾驶）本质上不可能精确求解，而深度学习恰恰擅长『不精确』的近似——这正是它适用面极广的原因。

> **imprecise** /ˌɪmprɪˈsaɪs/ *adj.*
> 不精确的

12:27

Almost right away, we started working on computer vision. Almost right away, we started working on robotics, self-driving cars because I that fundamental capability, you could imagine solving some imp- important problems in the area of computer vision and robotics. And so, so I think I think the big breakthrough was simply that this is much more foundational than AlexNet. This is a way of doing software. And the implications to the processor, the middleware, the algorithms, the applications, you know, what I now describe as the five-layer cake, that entire industrial stack, I imagine reinventing all together about 15 years ago. And this is simply about asking questions, reasoning about things to first principles, asking, you know, questions like, "If this, then what?" if this can get better, then so what? You know, asking all of the basic questions about something that you observe that's really impactful.

几乎是立刻，我们就开始做计算机视觉。几乎是立刻，我们就开始做机器人、自动驾驶汽车，因为有了那项基础能力，你可以想象在计算机视觉和机器人领域去解决一些重要问题。所以我认为，真正的大突破就在于：这件事比 AlexNet 本身要根基性得多。这是一种做软件的方式。而它对处理器、中间件、算法、应用的影响，也就是我现在说的「五层蛋糕」，整个这套产业堆栈，我在大约 15 年前就设想把它们全部重新发明一遍。这其实就是不断提问、按第一性原理去推理，去问「如果这样，那会怎样？」如果这个还能变得更好，那又意味着什么？就是对你观察到的、真正有影响力的事情，去问所有最基本的问题。

『五层蛋糕』指处理器、系统、中间件、算法、应用五层产业堆栈。方法论是第一性原理提问：『如果这个成立，接下来会怎样？』15 年前的推演决定了今天的布局。

foundational /faʊnˈdeɪʃənl/ *adj.*
基础性的，根本的

middleware /ˈmɪdlweɪr/ *n.*
中间件（连接系统与应用的软件层）

first principles *phr.*
第一性原理（从最基本事实出发推理）

06 创始人模式与冲浪式领导

13:30 Garry Tan

>> I mean, one of the things that really jumps out at me is to what degree you go all the way into the weeds. You read papers, you know, talk directly to the principal scientists who are sort of coming up with these things. Do you have any advice for people in the audience? I mean, that's like true founder mode. And then at the same time, you probably you have an organization and you have executives and you have people who say like, here's the graph, we want to stay on this graph. You know, sometimes it ruffles feathers. Like, do you have any advice for people about an organization and how you navigate that really? Like, how do you build an org that allows you to think in first principles? Cuz if the Fortune 500 did that, like, the Fortune 500 will probably look a lot more like Nvidia than not. And it doesn't. Like, you have built a very unique company.

>> 我觉得特别让我印象深刻的一点，是你会深入到多细的细节里。你会读论文，会直接跟提出这些东西的首席科学家交流。你对在座的人有什么建议吗？我是说，这才是真正的「创始人模式」。但与此同时，你又有一个组织，有高管，有人会说：这是我们的曲线，我们想沿着这条曲线走。有时候这会得罪人。你对如何管理一个组织、如何处理这种张力有什么建议吗？比如，怎么建立一个能让大家用第一性原理思考的组织？因为如果《财富》500 强都这么做，那它们看起来会更像英伟达，但事实并非如此。你确实建了一家非常独特的公司。

founder mode（创始人模式）：Paul Graham 2024 年提出的概念，指创始人跳过层级、深入细节地经营公司，与传统『授权式管理』相对。主持人问的是如何把它组织化。

into the weeds *phr.*
深入琐碎细节（go all the way into the weeds）

ruffles feathers *phr.*
惹恼人，得罪人（ruffle sb's feathers）

14:18 黄仁勋

>> My state of mind when I'm my state of mind is always starts with curiosity. I have a whole bunch of questions myself. And of course like anybody else, I'll seek the shortest path to the answer. But often times the answers from the people that are near me might not be satisfying and I might have other questions and maybe they're busy doing something and they're pursuing something. And so my first my first inclination is to go discover the answers to my own curiosity. My second is if I find that the information is in that the domain of information or you know, particular field could be really important to somebody and could be important to our company, then my next inclination is how can I learn as much as possible so that I could be of service to the company and share with the everybody else.

>> 我的心态，我的心态永远是从好奇心开始的。我自己脑子里就有一大堆问题。当然，和其他人一样，我会去找通往答案的最短路径。但很多时候，身边的人给出的答案可能并不能让我满意，我可能还有别的问题，而且也许他们正忙着做别的事，正在追求别的目标。所以我的第一反应就是自己去找出这些好奇心的答案。第二点是，如果我发现这些信息、这个信息领域，或者说某个特定领域，可能对某些人真的很重要，也可能对我们公司很重要，那我接下来的反应就是：我怎么才能尽可能多地学到东西，这样我就能为公司服务，并且分享给其他所有人。

黄仁勋的学习路径：好奇心、自己找答案；若发现该领域对公司重要，就系统性深挖并分享给全公司。学习被定义为 CEO 的服务职能，而非个人爱好。

inclination /ˌɪnkləˈneɪʃən/ *n.*
倾向，意愿（first inclination 第一反应）

be of service to *phr.*
为……效力，对……有用

15:13

You know, this is no different than you when you're sharing knowledge. I mean, I watch your podcasts and I watch your videos and I really enjoy them. You're sharing ideas with everybody else. You know, in a lot of ways I think a CEO is in service of the company, in service of all the people that are working there. And you want to empower them with some insight. And so that's really where it's coming from. It's not so much a management technique, but a personality technique. You know, I want to empower you and this is something really important that I just observed. Let me tell you why it's so important.

你看，这跟你分享知识没什么两样。我是说，我会看你的播客，也会看你的视频，我真的很喜欢。你也是在跟大家分享想法。你知道，在很多方面，我觉得CEO是在服务这家公司，服务所有在这里工作的人。你希望能给他们一些洞见，赋予他们力量。所以这其实就是它的来源。这与其说是一种管理技巧，不如说是一种性格上的方式。你知道，我想赋能给你，而这是我刚刚观察到的、非常重要的一件事。让我告诉你为什么它这么重要。

『CEO 是为公司和员工服务的』——与『CEO 发号施令』的常识相反。他把深入细节归因于个性而非管理技巧，暗示方法难以照搬，但心态可以学。

empower /ɪmˈpaʊər/ *v.*
赋能，使有能力

15:46

Now part of part of having to be near the ground and be in the weeds if you will, is because often times the technology is complicated or it's changing really fast. And especially when it's changing fast like our world, unless you have a tactile sensation of what is actually happening, it could either to you feel like it's just moving way too fast to understand. But if you understand the first principles of it over time, then everything kind of makes sense. You know, it's kind of like surfing I would imagine. I don't know how to surf, but I can imagine it's kind of like surfing. You get out on the wave. To me it looks like chaos, but to a surfer, you know, somehow they get right? They can read the waves and they know how to stay on top of it. And so I think being CEO is very similar to that. You know, you have to learn how to surf and order to learn how to surf you have to understand the waves. You have to be able to read the wind and you have to have good timing and you can't have any

那么，之所以必须贴近一线、深入细节，部分原因是技术往往很复杂，或者变化得非常快。尤其是当它变化得像我们这个世界这么快的时候，除非你对正在发生的事有一种切身的触感，否则你可能会觉得它快得根本来不及理解。但如果你随着时间理解了它的第一性原理，那一切就都说得通了。你知道，我想这有点像冲浪。我不会冲浪，但我能想象那有点像冲浪。你划出去迎浪。在我看来那就是一团混乱，但对冲浪者来说，他们不知怎么就能做到，对吧？他们能读懂海浪，知道怎么稳稳地站在浪头上。所以我觉得做CEO跟这个非常像。你得学会冲浪，而要学会冲浪，你就得理解海浪。你得能读懂风向，还得有很好的时机感，而这些你都不可能拥有，

冲浪比喻：外行看浪是混乱，冲浪者能『读浪』。对应观点：只要掌握第一性原理并保持一线触感，快速变化的技术就是可读的，而不是可怕的。

tactile /ˈtæktaɪl/ *adj.*
触觉的（tactile sensation 切身的触感）

stay on top of *phr.*
掌控，驾驭（局面）；字面为稳站在……之上

16:45

Of that unless you try unless you actually do it. And so partly is to inform myself, partly is to try to figure out, you know, what is try to break down the problem so that the company can learn it in a way that they can do something about. part of it is about inspiring other people. And you know, it's all those basic traits of all the people in this room. You don't have to change your personality or your behavior when you become CEO. It is possible for you to continue to be yourself.

除非你去尝试，除非你真的去做。所以一部分是为了让我自己了解情况，一部分是为了弄清楚——去把问题拆解开，让公司能以一种真正能采取行动的方式去理解它。还有一部分是为了激励其他人。而且你知道，这些都是在座各位本身就具备的基本特质。你不需要在成为CEO时改变自己的性格或行为方式。你完全可以继续做你自己。

要点：当 CEO 不需要改变个性。这是对台下年轻创业者的宽慰——不必模仿教科书式的领导者形象，做自己即可。

traits /treɪts/ *n.*
特质，品性

17:17

And one of the things that I that I learned a long time ago and I have no idea where I saw this. but you know, the CEO or the founders you are the you're building a car that you are going to race. You're going to build an F1 racer, but you're going to build it in a way that you can drive. You should adapt the car to you. You know, somebody I think had asked me you know, Jenson, if you if you don't use conventional management techniques and organizational techniques you know, what's going to happen when you leave the company?

还有一件事是我很久以前学到的，我完全想不起来是在哪儿看到的。但你知道，作为CEO或者创始人，你是在造一辆将由你自己去比赛的车。你要造一台F1赛车，但你得按照你自己开得动的方式去造它。你应该让车来适应你。我记得有人问过我：黄仁勋，如果你不用传统的管理方法和组织方法，那等你离开公司之后会怎么样呢?

F1 比喻的核心：创始人是车手，公司是赛车，应『让车适配车手』。这与主流管理学『建立不依赖个人的制度』的教条正好相反。

17:59

Well, you know, when I die on the job someday you know, I told them they'll just have to reshape the company for the next CEO. And the reason that's wisdom is because we're the F1 drivers. You know, we're the racers. And the world is really competitive and we've got to stay we've got to you know, we've got to win. And we've got to achieve our mission. And so whatever it takes to fit the car to you whatever it takes to fit the organization to you, that's what you got to do. And the next CEO, whatever the personality is, they can figure it out.

我跟他们说，等我哪天在岗位上去世了，他们只需要为下一任CEO重新塑造这家公司就好。这话之所以是智慧，是因为我们才是F1车手。我们是赛车手。而这个世界竞争非常激烈，我们必须保持——我们必须，你知道，我们必须赢。我们必须完成我们的使命。所以，不管要做什么才能让车适合你，不管要做什么才能让组织适合你，那就是你必须去做的事。至于下一任CEO，不管他们是什么性格，他们自己能想明白。

更反常识的继任观：不为接班人预留制度，『下任 CEO 上任时重塑公司即可』。理由是竞争激烈，当下赢下比赛比制度的延续性更优先。

whatever it takes *phr.*
不惜一切代价，无论付出什么

18:32

>> Amazing. I mean, that it does seem like any change you make to the car will just slow you down and lose you races that you know, isn't fit to you. >> Yeah, or we're constantly tweaking the car to our needs. And I'm that's really what I'm doing all the time. I'm constantly tweaking the company, constantly reshaping business processes and the way things work so that I can you know, be more effective for the company. >> True founder mode. >> Yeah, founder mode. Founder mode could scale for 34 years.

>> 太精彩了。我是说，确实，任何对车做的、不适合你的改动，都只会拖慢你的速度，让你输掉比赛。你知道，那些不适合你的改动。>> 是啊，或者说我们一直在根据自己的需要微调这辆车。而这其实就是我一直在做的事。我一直在微调这家公司，不断重塑业务流程和运作方式，好让我能为公司发挥更大的效力。>> 真正的创始人模式。>> 是的，创始人模式。创始人模式可以扩展34年。

1993 年创立至今约 33 年，『创始人模式可以规模化运转 34 年』——以自身为反例，回应『创始人模式无法持久』的质疑。

tweaking /ˈtwiknŋ/ *v.*
微调，小幅调整（tweak）

07 智能体：新软件与可控性

19:02 Garry Tan

>> That's right. >> From zero to 5 trillion. No evidence. No >> [applause] >> I'd love to switch gears to like what you know, what are the what are the frontier algorithms that you're most interested in now? I mean I love that you're all the way down into the material science, all the way up into the app level. you know, you're the first to speak on stage about Open Claw and now Hermes agent. I wonder if you can sort of like walk us through a day in the life of like how you think about the different stages. I mean, going from materials to chips to data centers to even like the app level, like how people are going to work. Like there's sort of this idea of a full stack AI factory.

>> 没错。>> 从零做到5万亿。没有反例。没有 >> [掌声] >> 我想换个话题，聊聊你现在最感兴趣的前沿算法是什么？我是说，我很喜欢你既能一路深入到材料科学，又能一路向上到应用层。你知道，你是第一个在台上谈到 Open Claw、现在又谈到 Hermes agent 的人。我想请你带我们过一遍你的一天，看看你是怎么思考这些不同层级的。我是说，从材料到芯片，到数据中心，甚至到应用层，到人们将来会怎么工作。有点像那种全栈AI工厂的概念。

主持人抛出『全栈 AI 工厂』概念：从材料、芯片、数据中心一直到应用层。Open Claw 与 Hermes 是访谈语境中的开源智能体项目。

switch gears *phr.*
换挡；（喻）转换话题或方向

frontier /frʌnˈtɪr/ *adj./n.*
前沿（的）；边界

19:50 黄仁勋

>> Well, this is one of the things that is probably going to be the most useful skill in the future. And in fact, just in listening to you talk about technology and you your use of it, you know, one of the most important things is systems understanding. Systems awareness, system design, system organization. but systems thinking. And the reason for that is because most of the low-level things that has to be done are going to be done agentically anyways. They're going to be automated anyhow. And so, whether it's, you know, in my generation it's about compiling chips and synthesizing transistors and gates and functional blocks and all of that is now synthesized. And so, most of our designers are systems designers.

>> 嗯，这可能是未来最有用的一项能力。事实上，光是听你谈技术、谈你怎么使用技术，你知道，最重要的东西之一就是对系统的理解。系统意识、系统设计、系统组织。嗯，也就是系统思维。原因在于，大部分必须完成的底层工作，反正都会由智能体来做，反正都会被自动化。所以，你看，在我那一代，工作是编译芯片、综合晶体管、门电路和功能模块，而这些现在都已经是自动综合出来的了。所以我们大多数设计师都是系统设计师。

核心判断：底层执行（写代码、综合电路）将由智能体自动完成，人的价值上移到系统层面。芯片设计早已如此——晶体管、逻辑门由 EDA 工具自动综合，设计师都是系统设计师。

agentically /eɪˈdʒɛntɪkli/ *adv.*
以（AI）智能体方式地（AI 行业新词）

synthesizing /ˈsɪnθəsaɪzɪŋ/ *v.*
综合；（芯片设计）自动综合生成电路

20:38

In the case of software, most software is going to be done agentically anyhow. So, you have to be much more able to think abstractly about systems. What are the what are the problems you're trying to solve? What are the constraints? Where, you know, where's the input? Where's the output? You know, where information coming from? what is the rate of information flowing in and out of the system? what are the constraints? you know, and so is it processor? Is it memory? Is it networking? you know, and so under these systems problems at a sufficiently technical level is going to be very helpful to all of the people in this room. And I don't think that way of that fundamental knowledge is ever going to be useless. I think it's going to be more and more useful. And so I try to understand systems I the best I can. One of the things One of the things that speaking of agent, the fact of the matter is we kind of have coarse level recursive self-improvement already.

在软件方面，反正大部分软件也都会由智能体来完成。所以你必须更能抽象地思考系统。你要解决的问题是什么？约束条件是什么？输入在哪里？输出在哪里？信息从哪里来？信息进出系统的速率是多少？约束是什么？是处理器的问题？是内存的问题？还是网络的问题？所以，在足够技术性的层面上理解这些系统问题，对在座各位都会非常有帮助。我不认为这种思考方式、这种基础知识会有变得没用的一天。我觉得它只会越来越有用。所以我尽我所能去理解系统。说到智能体，有一件事——事实上，我们其实已经有了粗粒度的递归自我改进。

『粗粒度递归自我改进』：智能体每次使用后更新自身配置与记忆，越用越聪明。递归自我改进原是 AGI 理论中的概念，此处指它已在工程上初步发生。

abstractly /æbˈstræktli/ *adv.*
抽象地（think abstractly 抽象思考）

recursive self-improvement *phr.*
递归自我改进（AI 通过改进自身而加速变强）

coarse /kɔrs/ *adj.*
粗糙的；粗粒度的（与 fine-grained 相对）

21:44

And the fact that every time you use it, it improves the markdown files. every time you use it, it updates its long-term memory. And the long-term memory is being processed either compacted or turned into knowledge graphs or, you know, so on and so forth. it's being improved all the time. you know, asynchronously. And so the agent's getting smarter every time. Still, the problem is and this is one of the one of the problems that I think it'd be helpful for everybody to solve is how can we have very specific fine-grained control?

每次你用它，它都会改进那些 markdown 文件。每次你用它，它都会更新长期记忆。而长期记忆又会被进一步处理，要么被压缩，要么被转化成知识图谱，等等等等。它一直在被改进。而且是异步进行的。所以智能体每一次都在变得更聪明。不过问题还是在于——这也是我认为大家如果能解决会很有帮助的问题之一——我们怎样才能做到非常非常具体的细粒度控制？

智能体长期记忆的工程手段：压缩、转成知识图谱、异步后台处理。黄仁勋点出的开放难题是『细粒度控制』，下一段展开。

fine-grained /ˈfaɪn greɪnd/ *adj.*
细粒度的，精细的

asynchronously /eɪˈsɪŋkrənəsli/ *adv.*
异步地（不阻塞主流程地后台进行）

22:18

You know, if not for rags, if not for conditional inputs, if not for our all of our prompts directly into output was too coarse. And so the fact that we can condition, the fact that we can control the agents all the way down to eventually when it comes up with a plan, I change one word in a plan file, and that one word makes a delta difference. Not complete difference, but specific difference. maybe it's one pixel, maybe it's one triangle, maybe it's one component in a CAD file, maybe one layer, one via, one connection.

你知道，如果不靠RAG，不靠条件输入，不靠我们所有的提示词，直接从输入到输出就太粗糙了。所以，我们能够施加条件、能够控制智能体，一直细到最终——当它给出一个方案时，我在计划文件里改一个词，那一个词就带来一个增量式的差别。不是完全不同，而是特定的差别。也许是一个像素，也许是一个三角面，也许是CAD文件里的一个元件，也许是一层、一个过孔、一条连线。

RAG（检索增强生成）是给模型外挂知识库的技术。理想的控制粒度：在计划文件里改一个词，只产生对应的增量变化（一个像素、一个零件、一个过孔），其余自动重生成。

delta /ˈdɛltə/ *n.*
增量，差值（工程用语，makes a delta difference）

via /ˈvaɪə/ *n.*
（电路板、芯片中的）过孔；此处非介词用法

22:57

And then it regenerates everything else. I think that level of control and that level of collaboration with agents will be game changing. We don't need the agents to be 100% accurate, 100% high quality in order for us to use it. It could, you know, literally be 80% and then we help it the rest of the way, or it could be 99% we help it the rest of the way. And so I think controllability is probably the single biggest breakthrough that we need for agents at every single level. >> Do you think people will like I mean, with Hermes or Open Claw, it feels like that might actually be somewhat existential. Like people should control their own personal AGI. Like they shouldn't outsource that app and, you know, have it be just in the cloud and someone else's agent that like kind of tells you what to do. Like you kind of want it to be your own.

然后它把其余的部分重新生成。我认为这种程度的控制、这种程度的人机协作会是颠覆性的。我们并不需要智能体做到100%准确、100%高质量才能用它。它可以只做到80%，剩下的我们来补；也可以做到99%，剩下的我们来补。所以我认为可控性大概是我们在每一个层面上都最需要的那个突破。每一个层面。>> 你觉得人们会……我是说，看到 Hermes 或 Open Claw，感觉这件事其实可能带有某种存亡意味。就是人们应该掌控属于自己的个人AGI，不应该把那个应用外包出去，让它只跑在云端，变成别人的智能体，反过来告诉你该做什么。你会希望它是属于你自己的。

重要且反直觉：智能体不需要 100% 准确才能用，80% 或 99% 由人补齐即可。因此他认为『可控性』（而非准确率）才是智能体最需要的单项突破。

game changing *adj./phr.*
颠覆格局的，革命性的

existential /ˌɛgzɪˈstɛnʃəl/ *adj.*
关乎存亡的；存在层面的

outsource /ˈaʊtsɔrs/ *v.*
外包

23:47

>> Yeah. Is that part of the thrust behind Nvidia being so involved in >> I think well, first of all, I need to understand agents because agents is the new software. And how is this new software processed matters a lot >> to computer architecture. >> And the more intimate we are about the nature of agents and how it's different than chatbots, which is how different than maybe inference in the very beginning. However, we think about these processing layers, the more intimate we are about the nature of the processing, the better we can design systems.

>> 是的。这是不是英伟达如此深度参与其中的部分动因？ >> 我想，首先，我需要理解智能体，因为智能体就是新的软件。而这种新软件是怎么被运行处理的，对计算机架构来说非常关键。>> 而我们越是深入了解智能体的本质，了解它和聊天机器人有何不同、和最初的推理有何不同——不管我们怎么划分这些处理层次——我们越是深入了解处理的本质，就越能设计出好的系统。

『智能体是新的软件』：新软件的处理方式（推理、记忆、工具调用）决定未来计算机体系结构的设计方向——这是英伟达必须深入理解智能体的第一个原因。

intimate /ˈɪntəmət/ *adj.*
熟稔的，有深入了解的（be intimate about sth）

08 活在未来的系统设计法

24:24

We kind of have to live in the future 5 to 10 years because it takes three or so years just to build a system, takes a couple years to ramp it up, and you're dealing and you would like them to be able to use the computer for 10 years after. And so you kind of have to live in the future for a while. And so agentic systems for us at the first principles is just what is the workload, what's the algorithm, how is it going to evolve, where are the bottlenecks, you know, where are the Amdahl's law's problems, and how it scale, what happens to concurrency? How do you deal with sandboxes?

我们某种程度上必须活在5到10年之后的未来，因为光是造出一套系统就要三年左右，再花一两年爬产，而且你还希望客户在那之后还能用这台计算机用上10年。所以你得先在未来生活一段时间。因此，对我们来说，智能体系统的第一性原理就是：负载是什么，算法是什么，它会怎么演进，瓶颈在哪里，阿姆达尔定律的问题出在哪里，它怎么扩展，并发会发生什么变化？沙箱怎么处理？

芯片系统从设计到量产约需 3 年加数年爬坡，客户还要使用近 10 年，所以必须『活在未来 5 到 10 年』。阿姆达尔定律：系统加速上限受不可并行部分制约，是体系结构的经典约束。

ramp it up *phr.*
爬坡提升；逐步扩大产量（ramp up）

bottlenecks /ˈbɑtlnɛks/ *n.*
瓶颈

concurrency /kənˈkɜrənsi/ *n.*
（计算）并发，并发性

sandboxes /ˈsændbɑksɪz/ *n.*
沙箱（隔离的安全运行环境）

25:01

How do you deal with MCP? How do you deal with you know, working memory, long-term memory? How do you have all these autonomous systems, asynchronous systems working all the time? And so, what kind of design architecture makes perfect sense for that? And so, we have to go and go discover that. And then, of course, the second thing is I want to use agents ourselves to make NVIDIA go faster. And so, we have, you know, voices in the back and we've got cloud code autonomously running in sandboxes all over NVIDIA, and that's really fantastic.

MCP怎么处理？工作记忆、长期记忆怎么处理？怎么让这些自主系统、异步系统持续不断地运转？那么什么样的设计架构才最契合这一切？这些我们必须自己去探索。当然，第二件事是，我希望我们自己也用智能体来让英伟达跑得更快。所以我们后台有各种声音，我们在英伟达各处的沙箱里自主运行着 Claude Code，这真的非常棒。

MCP（Model Context Protocol）是连接智能体与外部工具的开放协议。英伟达内部让 Claude Code 等编码智能体在沙箱中自主运行——第二个原因：用智能体加速自己。

autonomous /ɔˈtɑnəməs/ *adj.*
自主的，自治的

25:31

And some people use code codex, some people use cloud code, some people use cursor, some people use cognition. And we let kind of a thousand flowers bloom, let people select the tools they want to use, and then we learn from all of that. And so, the second part is just helping the company move faster. Use the tools, and the more they use it, the more you're going to learn about how to make it work better in the future. And then the last part is discovering the future of solutions technology for the future. And maybe you know, when we when we saw when we saw the early versions of chain of thought come out of Stanford, it was probably a decade ago at this point, maybe eight years ago.

有些人用 Codex，有些人用 Claude Code，有些人用 Cursor，有些人用 Cognition。我们让百花齐放，让大家自己选择想用的工具，然后我们从中学习。所以第二部分就是帮助公司跑得更快。去用这些工具，用得越多，你就越能学到将来怎么把它做得更好。然后最后一部分，是去发掘未来的解决方案技术。也许，你知道，当我们看到当我们看到斯坦福最早那批思维链（chain of thought）的工作时，大概是十年前，也许八年前。

『百花齐放』策略：不统一指定工具，让员工自选 Codex、Claude Code、Cursor、Cognition 等，再从使用中学习。思维链（chain of thought）：让模型分步推理的技术。

a thousand flowers bloom *phr.*
百花齐放（let a thousand flowers bloom，允许多种做法并存）

chain of thought *phr.*
思维链（让模型分步推理的技术）

26:14

You know, the question is how effective is that going to be in reasoning, and how scalable is going to be? And what is the implication, for example, in computer vision, if we can reason from prior knowledge. And then the big breakthrough, of course, is just in thinking through that small little domain, you come to realize that maybe we don't need as much data for cars to train a self-driving car. Which led us to creating Alpaca My which is the world's first thinking self-driving car.

问题就是：它在推理上会有多有效？它的可扩展性如何？比如说，如果我们能基于先验知识进行推理，那对计算机视觉又意味着什么？然后，真正的大突破当然是：仅仅把那个很小的领域想透，你就会意识到，也许训练一辆自动驾驶汽车并不需要那么多数据。这就促使我们做出了 Alpaca My，世界上第一辆会思考的自动驾驶汽车。

推理链条示范：思维链、基于先验知识推理、训练自动驾驶所需数据大减，于是做出 Alpamayo（文中口误作 Alpaca My），号称第一辆『会思考』的自动驾驶车。

scalable /ˈskeɪləbl/ *adj.*
可扩展的，可规模化的

prior knowledge *phr.*
先验知识（事先已具备的知识）

26:50

And with just a million miles or so, a couple million miles, it's an incredibly great self-driving car. And the reason for that is it's kind of like us, right? We don't need that many miles before we could drive fairly well most of our lives. And the reason for that is because we have prior knowledge from our language model, and we can decompose a situation we've never seen before, and I and build it up out of things that we understood and know very well. And so, that's an example of seeing something and then realizing the impacts on sometime later. when the agentic systems came along, it's very clear that obviously a large language models needs memory, it needs prior knowledge, it needs tools, it needs ways to network with other agents. And so, that kind of, you know, that once you see some early indicators, and you're able to reason about the future, helps you get a leap, you know, into the future.

只用了大概一百万英里、两三百万英里的数据，它就是一辆极其出色的自动驾驶车。原因在于，它有点像我们人类，对吧？我们并不需要开很多英里，就能在大部分人生里开得相当不错。原因在于我们有来自语言模型的先验知识，我们能把一个从没见过的场景拆解开，再用我们已经理解、非常熟悉的东西把它重新搭建起来。所以这就是一个例子：你先看到某样东西，然后在一段时间之后才意识到它的影响。当智能体（agentic）系统出现时，非常非常清楚的一点是，大语言模型显然需要记忆，需要先验知识，需要工具，需要能与其他智能体互联的方式。所以说，一旦你看到一些早期的苗头，并且能够对未来做出推理，就能帮你向未来跃进一大步。

类比论证：人类只需很少驾驶里程就能开好车，因为有语言与世界的先验知识，能把新情境分解为已知要素。该车仅用一两百万英里数据——对比传统路线动辄数十亿英里。

decompose /ˌdikəmˈpoʊz/ *v.*
分解，拆解（新情境拆成已知要素）

09 开源信念与人人自建 AI

27:54 Garry Tan

>> I feel like there's this pattern that I'm starting to see around Nvidia. It's like you see a problem, there's a new algorithm, there's some new thing happening, and then actually you're right there with open source. I mean, I remember when OpenCL came out and people said it was unsafe, but you guys came out with sandboxing sort of toolkit that like surrounds any harness and makes it safe. And so, >> When I saw OpenCL, my first thought was Well, first of all, I learned about it. And then and then you know, without much imagination, you just realized we just designed the modern computer. This is the operating system that's going to hold a large language model.

>> 我感觉我开始看到英伟达身上的一种模式。就是你看到一个问题、出现一个新算法、出现某种新东西，然后你们总是和开源站在一起。我记得 OpenCL 出来的时候，有人说它不安全，但你们拿出了一套沙箱化的工具包，能把任何运行环境包起来，让它变得安全。所以说，>> 我看到 OpenCL 时的第一反应是……首先，我先去了解了它。然后，不需要太多想象力，你就会意识到：我们刚刚设计出了现代计算机。这就是那个用来承载大语言模型的操作系统。

主持人总结英伟达的行为模式：发现新算法就立即跟进并配套开源工具。注意：历史上的 OpenCL 是 2009 年的并行计算标准，此处语境指访谈中的开源智能体项目，属新的用法。

harness /ˈhɑrnɪs/ *n.*
挽具；（软件）执行框架、运行环境

28:33 黄仁勋

And in a lot of ways, OpenCL to me was very Linux moment to me. >> Yeah. >> And now everybody can build their own AI. And I was so excited about that. And we contacted Peter, and we said, "Hey, you know, all of Nvidia's engineers are your engineers. That's what I told Peter. You got this battleship outside your house. You know, break down the problem as you desire and we'll contribute as you wish. Same thing with the Hermes team. You know, and I'm so excited about the work that they're doing.

在很多方面，OpenCL 对我来说非常像一个 Linux 时刻。>> 是的。>> 现在每个人都能构建自己的 AI 了。我对此非常兴奋。我们联系了 Peter，我们说：「嘿，英伟达所有的工程师都是你们的工程师。」我就是这么跟 Peter 说的。你家门口停着这艘战舰。你想怎么拆解这个问题都行，我们会尽我们所能地贡献希望。Hermes 团队也是一样。你知道，我对他们正在做的工作感到非常兴奋。

『英伟达所有工程师都是你的工程师』『门口停着一艘战列舰』——向开源项目倾斜全公司资源的表态。类比 Linux 时刻：操作系统级的开放平台让所有人都能构建自己的 AI。

29:07

I do think that the world needs the ability for everybody to build their own AI. And you could you could of course and I encourage everybody to use cloud services as much as possible. Everybody should use chat GPT and Claude and right, everybody should use that. And but if you if you need to build your own AI because you're a company and you need to build your own domain specific AIs. Now you have Hermes and you have open Claude, you've got all kinds of you got LangChain, deep agent, you got all these different ways, right?

我确实认为，这个世界需要让每个人都有能力构建自己的 AI。当然你也可以——我也鼓励每个人尽可能多地使用云服务。每个人都该用 ChatGPT，用 Claude，对吧，大家都应该用。但如果你需要构建自己的 AI，因为你是一家公司，你需要构建自己领域专用的 AI。现在你有 Hermes，有 open Claude，还有各种各样的——你有 LangChain、deep agent，你有这么多不同的方式，对吧？

分层建议：个人尽管用 ChatGPT、Claude 等云服务；企业则需要用 Hermes、LangChain 等框架自建领域专用 AI。开放生态与商用云服务并不互斥。

domain specific *adj./phr.*
领域专用的（domain-specific AI 领域专用 AI）

29:40

To build your own AI. And it's quite frankly relatively easy because the software is smart. You know, and so AI smart and therefore AI must be so smart you could adapt it easily. And so I think that we want we want to encourage everybody and every company to build their own AIs. And who knows what innovation will come from the fact that it's open source. >> I feel like all the alpha is in building your own AI. I mean, if someone else is using whatever is off the shelf, but you're you have a thing that can recursively self-improve and it is, you know, I mean, the mech people are very flippant about market markdown files.

去构建你自己的 AI。而且坦白说，这相当容易，因为软件很聪明。你知道，AI 很聪明，所以 AI 一定聪明到你可以轻松地去适配它。所以我认为我们希望鼓励每一个人、每一家公司都去构建自己的 AI。而且谁知道呢，正因为它是开源的，会催生出什么样的创新。>> 我觉得所有的超额收益都在于构建自己的 AI。我是说，如果别人用的是现成的东西，而你手里有一个能够递归自我改进的东西，你知道，我是说，搞 MCP 的那些人非常不把 markdown 文件当回事。

alpha 是金融术语，指跑赢市场的超额收益。主持人的观点：真正的竞争优势在于自建可自我改进的 AI，而非人人可得的现成产品。

alpha /ˈælfə/ *n.*
（金融）超额收益；（喻）人无我有的优势

off the shelf *phr.*
现成的，即买即用的

flippant /ˈflɪpənt/ *adj.*
轻率的，不当回事的

30:18 Garry Tan

They say like, oh haha, it's just text, but like text is intelligence. And we're in a different >> Words are thoughts. >> Yeah. >> Yeah, words are thoughts. >> Yeah, and it turns out you can >> Try to try to think without words. >> Yeah, that's right. >> [laughter] >> So switching gears again, I mean, a lot of people are anytime you move the cheese, people get a little worried. Intelligence is going to be on tap, which is really awesome. I think it bodes well for everyone in this room. what do you think changes about the economy? What do you think, you know, happens in sort of a broader sense?

他们会说，哈哈，那不就是文本嘛，但文本就是智能啊。而我们处在一个不同的—— >> 语言即思想。>> 是的。>> 对，语言就是思想。>> 对，而且事实证明你可以—— >> 你试试不用语言去思考。>> 对，没错。>> [笑声] >> 那我们再换个话题。我是说，只要一动别人的奶酪，很多人就会有点担心。智能将会像自来水一样随取随用，这真的很棒。我觉得这对在座的每一位都是好兆头。嗯，你觉得经济会发生什么变化？你觉得，你知道的，从更宏观的层面上会发生什么？

『动了奶酪』化用畅销书《谁动了我的奶酪》，指变革引发不安；『智能随取随用』（on tap）像自来水一样打开即得——由此引出对经济影响的提问。

move the cheese *phr.*
动奶酪（喻打破现状令人不安，出自《谁动了我的奶酪》）

on tap *phr.*
（像自来水般）随取随用的

bodes well for *phr.*
对……是好兆头（bode well/ill for）

10 AI 消除任务而非岗位

30:51 黄仁勋

>> obviously, what I'm going to say is uneven. there are some you know, we're going to automate tasks. We're going to automate cognitive tasks. If that task is somebody makes a phone call and sends a bunch of words, you know, across the phone to you and your job is to provide a response. And if all the information is at your fingertip because you have all the database here and you should be able to answer that question completely. in that case, that task will be automated away. Okay? Ignoring that for a second. Not that Not that you Not that we ignore this, but my point is I'm going to answer the question about really the great opportunity. And so, many tasks will be automated away. many jobs Every single job will be will change and there'll be a whole bunch of new jobs and that I think we know. the bottom line is this.

>> 呃，显然，我要说的是：这个过程是不均衡的。呃，有一些，你知道，我们会把任务自动化。我们会把认知类的任务自动化。如果那个任务是，呃，有人打来电话，说了一堆话，通过电话传给你，而你的工作就是给出回应。而且，嗯，如果所有信息都在你手边，因为所有数据库都在这里，你应该能够完整地回答那个问题。呃，在这种情况下，那个任务就会被自动化掉。好吧？先把这个放一边。不是说你——不是说我们要忽略它，但我的意思是，我想回答的其实是关于巨大机会的那个问题。所以，嗯，很多任务会被自动化掉。嗯，很多工作——每一份工作都会改变，并且会出现一大批新的工作，这个我想我们都知道。嗯，归根结底是这样的。

回答的前提区分：会被自动化的是『凭既有信息即可完整作答』的认知任务（如接电话答疑）。注意他刻意先承认影响不均衡，再把话题引向机遇一侧。

uneven /ʌnˈivən/ *adj.*
不均衡的，不平均的

cognitive /ˈkɑgnətɪv/ *adj.*
认知的（cognitive tasks 认知类任务）

at your fingertip *phr.*
触手可及（常作 at one's fingertips）

31:46

The evidence would show that and it makes perfect sense that AI and automation is creating jobs everywhere. The narrative about AI destroying jobs is exactly backwards. AI eliminate tasks. AI automates tasks away. But it doesn't necessary doesn't necessarily eliminate jobs. And the reason for that is because the job of a person has a purpose and that purpose has many tasks. Some of those tasks could be automated away. Many of those tasks cannot be. And so, the evidence suggests that here we are, we've automated coding, which is a task, but the job of a software engineer appears to be growing, right? The number of software engineer jobs year over year has increased 10%.

证据会表明——而且这完全说得通——AI 和自动化正在到处创造就业。关于 AI 摧毁工作岗位的说法完全说反了。AI 消灭的是任务。AI 把任务自动化掉。但它不一定会消灭工作岗位。原因在于，一个人的工作是有目的的，而那个目的包含很多任务。其中一些任务可以被自动化，但很多任务不行。所以，证据表明，看看现在，我们已经把写代码这件事自动化了，写代码是一个任务，但软件工程师这份工作看起来还在增长，对吧？软件工程师的岗位数量同比增长了 10%。

全场最系统的论点：AI 消除的是任务（task）而非工作（job）——工作由多项任务组成，只有部分可自动化。证据：编码被自动化后，软件工程师岗位反而同比增长 10%（讲者口径）。

32:37

The task of reading radiology scans has been automated, but the number of radiology jobs has increased some 20% in the last several years, even though AI's taken over the whole field. And the reason for that is because the backlog of patients is incredibly high. Now doctors and hospitals could admit a lot more patients. In order to admit a lot more patients, you need more nurses, more radiologists. And so, the same thing with software. We hit the backlog of ideas, the backlog of ambition and aspiration is so high that if we can automate away the task of programming, we could hire more software engineers to do more things. We could be more ambitious.

阅读放射影像这个任务已经被自动化了，但放射科的岗位数量在过去几年里增长了大约 20%，尽管 AI 已经接管了整个领域。原因在于，积压的患者数量高得惊人。现在医生和医院可以收治更多的病人。而要收治更多病人，你需要更多护士、更多放射科医生。软件也是同样的道理。我们撞上了创意的积压，雄心和愿景的积压如此之多，以至于如果我们能把编程这个任务自动化掉，我们就能雇更多的软件工程师去做更多的事。我们可以更有野心。

放射科案例：AI 大量接管读片任务，但岗位数几年内增约 20%。机制：积压病例巨大，效率提升带来吞吐量增加，反而需要更多人——与杰文斯悖论（效率提升扩大总需求）同构。

radiology /ˌreɪdiˈɑlədʒi/ *n.*
放射学；放射科（读 X 光、CT 等影像）

backlog /ˈbæklɔg/ *n.*
积压（的工作、订单、病例）

33:23

Same thing all you know, just across the board. they said Harvey is going to eliminate all of the paralegal jobs, and the number of lawyers will be reduced. Turns out paralegals are growing like crazy. And the reason for that is because the backlog of lawsuits is really high, and now these law firms could get a lot more cases through. And in order to do so, you got to hire more people. And so, this is a classic example of productivity increasing growth. Increasing growth drives more employment. This is the reason why there's more employment today than there was when I first came out of school.

方方面面都是同样的道理。呃，他们说 Harvey 会把所有律师助理的工作都干掉，律师的数量会减少。结果律师助理增长得飞快。原因在于，积压的诉讼案件实在太多了，现在这些律所可以处理更多的案子。而要做到这一点，你就得雇更多的人。所以，这是一个非常经典的例子：生产力提升带来增长，而增长的提升带动了更多的就业。这就是为什么今天的就业岗位比我刚毕业时还要多。

Harvey 是面向律所的法律 AI 公司。『律师助理将消失』的预言落空，岗位反而大增——同一机制：积压诉讼被释放。结论：生产率带动增长，增长带动就业。

paralegal /ˌpærəˈligl/ *n.*
律师助理

across the board *phr.*
全面地，各领域无一例外地

11 物理 AI 与机器人时刻

34:00 Garry Tan

>> So, we've been talking a lot about software and agents. another really exciting thing that Nvidia is all the way out on the edge on is actually physical robots. you know, how far out? I think in the past you might have even said this as soon as this year. What's the latest thinking on, you know, when can we expect practical robotics? >> Yeah, the moment that I saw as generating video, that was that was a great moment for me. The moment that I and I start I saw as generating video, I mean, we did the original work on auto progressive GANs, okay? And we did the original work on conditional GANs. long before the first videos were generated outside that people saw, a couple of years earlier inside our labs, we were driving a simulator completely generated by video. And computer completely generated by neural networks. And so, the moment I saw us generating articulation, if I can generate video of a finger moving, if I could generate video of a hand picking up a glass, why can't I

>> 我们聊了很多软件和智能体。嗯，另一件非常令人兴奋的事情是，英伟达一直站在最前沿的其实是实体机器人。嗯，你知道，还有多远？我记得你过去甚至说过，可能就在今年。关于实用化的机器人什么时候能到来，你最新的看法是什么？>> 是的，我看到我们能生成视频的那一刻，对我来说是个非常重要的时刻。当我看到我们能生成视频的时候——我是说，我们当年做过渐进式 GAN 的原创工作，对吧？我们也做过条件GAN 的原创工作。嗯，在外界看到第一批生成视频之前很久，嗯，早了好几年，在我们的实验室里，我们就已经在驾驶一个完全由视频生成的模拟器了。完全由神经网络生成的。所以，当我看到我们能够生成关节运动的那一刻，如果我能生成一根手指移动的视频，如果我能生成一只手拿起玻璃杯的视频，为什么我不能

渐进式 GAN、条件 GAN 都是英伟达研究院 2017 年前后的成名工作。推理方式仍是类推：既然能生成手部动作的视频，就应能让机器人做出同样的动作。

articulation /ɑrˌtɪkjəˈleɪʃən/ *n.* 关节活动；（机器人）关节运动控制

35:10 黄仁勋

Cause a robot to do the same? And so, the moment I saw that generative AI happening, I realized that robotics articulation was around the corner. And so, now the question is, you know, how's the robot going to understand to generate motions that obey the laws of physics? How's it How does it understand causality? how does it understand, you know, friction, tension? How does it understand the laws of physics? And so, it started us down the journey of creating what we call physical AI now. And everybody calls it physical AI. And physical AI, we started working on world foundation model, an AI that understands the laws of physics and the how the world works. And we started down the journey of working on robotics. I would say the chat GPT moment of robots happened a couple of years ago already.

让机器人做同样的事？所以，当我看到生成式 AI 出现的那一刻，我就意识到机器人的关节运动近在眼前了。那么现在的问题是，你知道，机器人要怎么理解，呃，怎么生成符合物理定律的动作？它怎么理解因果关系？嗯，它怎么理解，你知道的，摩擦力、张力？它怎么理解物理定律？于是这就带着我们踏上了创造我们现在称之为物理 AI 的旅程。现在大家都叫它物理 AI。而物理 AI，呃，我们开始做世界基础模型，嗯，一种理解物理定律、理解世界如何运作的 AI。然后，嗯，我们踏上了做机器人的旅程。我认为机器人的 ChatGPT 时刻其实几年前就已经发生了。

『物理 AI』：理解物理定律、因果、摩擦并能在真实世界行动的 AI，核心是『世界基础模型』。断言『机器人的 ChatGPT 时刻几年前已发生』——比多数人的判断激进。

causality /kɔˈzæləti/ *n.*
因果关系

around the corner *phr.*
即将到来，近在眼前

36:01

>> Wow. >> And the reason for that is remember when chat GPT first came out, it didn't do anything productive. It didn't do anything useful, but it opened our imagination about what's possible. And I would say a couple of years ago, you know, robots walking around that we could do reinforcement learning, fine-tune it for and ground it in physics, really happened a couple of years ago. So, now what do we need to do? We need to do all the same things that we're doing now for agentic systems.

>> 哇。>> 原因在于，还记得 ChatGPT 刚出来的时候，它并没有做出任何有生产力的事。它没有做任何有用的事，但它打开了我们对于可能性的想象。我想说，几年前，你知道，机器人已经能走来走去了，我们可以做强化学习，为它做微调，把它锚定在物理规律上，这真的在几年前就发生了。那么现在我们需要做什么？我们需要做的，和我们现在为智能体系统做的事情完全一样。

类比的精确含义：ChatGPT 刚发布时并不『有用』，但打开了想象空间。机器人如今正处于同一阶段——能走、能做强化学习微调，尚未大规模产生经济价值。

reinforcement learning *phr.*
强化学习（通过奖励信号试错学习的方法）

fine-tune /ˌfaɪnˈtun/ *v.*
微调（在已训练模型上做针对性再训练）

36:32

We have to create environments for them to learn in, to eval in, eval against. And so, we have to do real to sim to create environments. we have to do we have to generate simulators that are based on simulation, grounded physics simulation, as well as generative physics simulations. And so, Isaac Sim, Cosmos, and all the work that we do in that area is related to simulation. And then the last part is sim to real. And so, that part is has something to do with reinforcement learning, grounding it on physics, grounding it on grounding it on all on all the electromechanical systems that robots require. And so, but these three basic system, I think builds up the eval, if you will, the post-training of robotics. And I think we're going to see it right around the corner.

我们必须为它们创造可以学习、可以评测、可以对照评测的环境。所以，我们必须做 real to sim（从真实到仿真）来创建环境。呃，我们必须，呃，我们必须生成基于仿真的模拟器，基于物理的仿真，以及生成式的物理仿真。所以，呃，Isaac Sim、Cosmos，以及我们在这个领域做的所有工作，都和仿真有关。然后最后一部分是 sim to real（从仿真到真实）。呃，这部分跟强化学习有关，嗯，把它锚定在物理上，锚定在，嗯，机器人所需要的所有机电系统上。所以，这三个基本的系统，我认为构建起了，呃，可以说是机器人的评测体系，以及后训练。我认为我们很快就会看到它了。

机器人训练三件套：真实到仿真（构建环境）、物理仿真与生成式仿真（Isaac Sim 是英伟达机器人仿真平台，Cosmos 是其世界基础模型系列）、仿真到真实（迁移回实体机器人）。

eval /ɪˈvæl/ *n.*
评测（evaluation 的 AI 行业缩略语）

37:24

>> Amazing. Where does physical AI show up first in a way that's really economically real? Are you seeing that already? >> We conjectured that robotics was going to come along and decided that the first application of robotics that has both a large enough market, relatively standardized technology so that we could scale and get the flywheel going, and has real economic value was self-driving cars. And so, inside Waymo, our chips from Nvidia. at Tesla, we were in the car. now we're in the data center. Mercedes, we're in the data center, we're in the car with a software stack.

>> 太棒了。物理 AI 最先会在哪里以真正有经济意义的方式落地？你已经看到这种迹象了吗？>> 我们当时推测机器人会到来，并且判断机器人的第一个应用——既有足够大的市场，嗯，技术又相对标准化，能让我们规模化、把飞轮转起来，嗯，同时还有真实的经济价值——就是自动驾驶汽车。所以，呃，Waymo 内部用的是我们英伟达的芯片。呃，在特斯拉，我们在车里。呃，现在我们在数据中心。嗯，梅赛德斯那边，我们在数据中心，也在车里，还带着一整套软件栈。

选自动驾驶作为物理 AI 首个落地场景的三条标准：市场够大、技术相对标准化（能形成飞轮）、有真实经济价值。Waymo 用英伟达芯片，特斯拉、奔驰在车端或数据中心侧有合作。

conjectured /kənˈdʒɛktʃərd/ *v.*
推测，猜想（conjecture）

flywheel /ˈflaɪwil/ *n.*
飞轮；（喻）自我强化的增长循环

38:07

We worked on Alpaca Myo, and we open-sourced it. And the reason why we open-sourced the self-driving car stack is because you need it for agriculture, you need it for mail delivery, you need it for warehouse AMRs. There's so many different ways that you could apply autonomous navigation and none of those markets are big enough to be a self-driving car market and we thought it was sufficiently diverse that we would create the whole stack for it. And so we're working with autonomous vehicles in all kinds of different places.

呃，我们做了 Alpamayo，并且把它开源了。我们之所以把自动驾驶汽车的软件栈开源，是因为农业需要它，邮件配送需要它，仓库的 AMR 需要它。有太多不同的方式可以应用，嗯，呃，自主导航，而这些市场中没有哪一个大到能撑起一个自动驾驶汽车市场，我们觉得它足够多样，值得我们为它做出整套软件栈。所以我们正在和各种各样的自动驾驶项目合作。

开源 Alpamayo 栈的商业逻辑：农业机械、邮件递送、仓储 AMR（自主移动机器人）等市场各自都不够大，但共享同一套自主导航技术——开源让长尾市场共同做大生态。

open-sourced /ˌoʊpən ˈsɔrst/ *v.* 将……开源（公开源代码）

38:38

Our robotics business, autonomous vehicle business, basically physical AI business is probably almost like $10 billion. So it's really, really big already. likely this will be one of the largest industries in the world and it'll take longer than a couple two, three years. It'll take less than 10. And so this will this will be our next $100 billion business. >> Amazing. I want to take a moment. I think this is the exact right crowd to you know, maybe as a arena we can welcome Jensen to X.

我们的机器人业务、自动驾驶业务，基本上就是物理 AI 业务，大概已经差不多有 100 亿美元的规模了。所以它已经非常非常大了。嗯，这很可能会成为世界上最大的产业之一，而且，嗯，呃，它需要的时间会超过两三年。但会少于 10 年。所以这将会是我们下一个千亿美元级的业务。>> 太棒了。嗯，我想花一点时间。呃，我觉得这里正好是最合适的一群人来，呃，你知道，作为一个舞台，我们可以欢迎 Jensen 加入 X。

量级披露：机器人加自动驾驶等物理 AI 业务已接近年 100 亿美元规模，预期成为『下一个千亿美元业务』，时间判断在 3 到 10 年之间。

39:10

Welcome to X. I mean, you made your first post and thank you for your leadership. >> [applause] >> You know, that's that just that shows you how introverted I am. It took me until 2026 to have the first post on X. You know, it's I'm probably the last human on Earth that did it. but what I posted was too important to me and too important to the to the industry and too important to the world. And so I over overcame my shyness and put my first thing out on X.

欢迎来到 X。我是说，你发了第一条帖子，呃，也感谢你的引领。>> [掌声] >> 你知道，这恰恰说明了我有多内向。一直到 2026 年我才在 X 上发了第一条帖子。你知道，我大概是地球上最后一个这么做的人了。呃，但是，呃，我发的那条内容对我来说太重要了，对这个行业太重要了，对这个世界也太重要了。所以，呃，我克服了自己的，嗯，我的羞怯，在 X 上发出了第一条内容。

自称『可能是地球上最后一个发 X 帖的人』（2026 年才发首帖）——为下文做铺垫：这件事（为开源发声）重要到让他克服内向。

introverted /ˈɪntrəvɜrtɪd/ *adj.*
内向的

39:47

>> No, thank you for your leadership. I mean, open source, open weights, open source models are incredibly important for I mean, what all of us in this room want to do. Like we want to create products. >> If not for open source, the mobile cloud industry would have never happened. If not for open, if not for Linux, if not for Kubernetes, if not for all of these, you know, platform, if not for TensorFlow or more important, PyTorch, right? The and the early versions of a cafe, right? Torch. I mean, all of the Theano. Remember the early versions of all Those were all open source. If not for all of that, how would we have modern AI?

>> 不，真的感谢你的引领。我是说，开源、开放权重、开源模型，对于我们在座所有人想做的事情来说，都极其重要。比如我们想创造产品。>> 如果没有开源，移动云产业根本就不会出现。如果没有开放，如果没有 Linux，如果没有 Kubernetes，如果没有所有这些，你知道的，平台，如果没有呃，TensorFlow，或者更重要的，呃，PyTorch，对吧？还有早期版本的 Caffe，对吧，Torch。我是说，所有这些Theano。还记得所有那些早期版本吗？那些全都是开源的。如果没有这一切，我们怎么可能拥有现代的 AI？

一份开源谱系：Linux（操作系统）、Kubernetes（容器编排）、PyTorch、TensorFlow、Caffe、Torch、Theano（深度学习框架）。论点：现代 AI 的每一层都建立在开源之上。

open weights *phr.*
开放权重（公开模型参数供下载使用）

12 给年轻人：学什么与韧性

40:26 Garry Tan

>> Well, thank you for your leadership and your voice is incredibly important here. Thank you. >> [applause] [applause] >> Before we go, I feel like we I just really resonate with your story. I think that everyone here, but I mean, would love the wisdom of, you know, your journey coming here. I mean, what should a young person learn now, given all the things that you're seeing, all the algorithms that are going to take hold in society? what should a young person learn now that will still matter, based on what you're seeing?

>> 嗯，感谢你的引领，你的声音在这件事上真的非常重要。谢谢。>> [掌声][掌声] >> 在结束之前，我觉得我们——我真的特别能共鸣你的故事。我想在座的每个人，我的意思是，都很想听听你一路走来的智慧。基于你所看到的这一切，所有这些即将在社会中扎根的算法，年轻人现在应该学些什么？嗯，基于你所看到的，年轻人现在应该学些什么，才是将来依然有价值的？你看到的？

41:01 黄仁勋

>> Well, some of the things that I saw today and some of the starters I met today was really quite encouraging and the thing that the big takeaway is, of course, the simple stuff is going to get automated away. And when I say simple stuff, I mean, software, you know, coding. the idea that you would you would do a you would solve a problem by sitting in front of a computer and you're actually writing, you know, writing code, that concept is obviously going to get automated away.

>> 嗯，我今天看到的一些东西，还有我今天见到的一些创业者，真的真的非常非常鼓舞人心。而且，嗯，最大的收获当然是，简单的事情终将被自动化掉。我说的简单的事情，是指软件，你知道的，写代码。呃，那种坐在电脑前解决问题、你实际在敲代码的方式，那个概念显然会被自动化掉。

给年轻人答案的前半：『简单的东西』会被自动化——包括坐在电脑前手写代码这一行为本身。注意这出自靠 AI 算力获利最多的公司 CEO 之口，分量不同。

takeaway /ˈteɪkəweɪ/ *n.*
核心收获，要点（the big takeaway 最大心得）

41:35

You know, in my generation, when I was when I was growing up, we had to do long division. I mean, for God's sakes, who has to learn long division, you know? And so, that got coded away, that got automated away. And so, I think the simple stuff is going to get automated away, but the hard problems, the hard sciences, physics, chemistry, biology, you know, computer science, computer engineering, systems thinking, you know, all and particularly the domains that are intersecting, those hard problems will never go away. And so, AI is just an incredible tool that helps us become even more ambitious.

嗯，你知道，在我们那一代，我小时候还得学长除法。我的天，现在谁还需要学长除法，对吧？所以那个东西被代码取代了，被自动化掉了。所以我觉得简单的事情会被自动化掉，但是那些难题、硬科学，嗯，物理、化学、生物，呃，还有计算机科学、计算机工程、系统性思维，呃，所有这些，尤其是各个领域交叉的地方，呃，那些难题永远不会消失。所以 AI 只是一个不可思议的工具，帮我们变得更有野心。

长除法类比：曾是必修技能，被计算器『自动化』后无人惋惜。持久的是难题与硬科学：物理、化学、生物、计算机科学与工程，尤其是学科交叉地带。

long division *phr.*
长除法（笔算多位数除法）

intersecting /ˌɪntərˈsɛktɪŋ/ *adj.*
交叉的，相交的（intersecting domains 交叉领域）

42:14

Even more impatient about solving these extraordinarily large and incredibly hard problems than before. And so, you know, if you if you look at my generation, when I first graduated, a chip designer would design a chip with maybe a thousand transistors, and that would be a very large chip. You know, now designing a trillion transistor chips is not even, you know, if somebody would have told me, "Jensen, our next chip is a trillion transistor." I said, "Okay." You know, it's not a thing. And the reason for that is because we are so ambitious now, the scale of the problem, the scale of the task is no longer a matter.

更加，嗯，更迫不及待地去解决那些比以往规模更大、难度更高的问题。所以，你知道，如果你看我们那一代，我刚毕业的时候，一个芯片设计师设计的芯片可能只有一千个晶体管，而那已经算是非常大的芯片了。你知道，现在设计万亿晶体管的芯片都不算什么了，如果有人跟我说：“Jensen，我们下一款芯片是一万亿个晶体管。”我会说：“行啊。”你知道吗，这都不算个事。原因就在于我们现在如此有野心，问题的规模、任务的规模已经不再是个问题了。

数量级对比：他毕业时设计千级晶体管的芯片已算大，如今万亿晶体管不足为奇。含义：工具把『规模』从约束变成常量，剩下的唯一约束是雄心。

42:55

And so, you don't have to worry about, you know, how much coding, how many engineers. You don't have to You don't have to think about those things anymore. You just have to think about what is the what is the problem you have to solve. And so, I think that the deep tech stuff, the deep science stuff, understanding the intersection between technology and social issues, understanding market gaps and holes, opportunities, I think all of that still exists.

所以你不用去担心要写多少代码、要多少工程师。你不用去想那些事情了。你只需要去想，你要解决的问题到底是什么。所以我觉得那些深科技的东西、深科学的东西，呃，理解技术与社会议题之间的交叉点，嗯，理解市场的空白和缺口，呃，理解机会，我觉得所有这些依然存在。

值得学的清单后半：技术与社会议题的交汇、市场缺口与机会的识别——执行可以被自动化，『发现值得解决的问题』仍然是人的工作。

43:23

And the better you are at systems thinking so that you could orchestrate millions of agents solving problems autonomously, the better off you are. And so, that's why system thinking is going to be so important. But otherwise, I think the world's going to continue to have a lot of great challenges for us to solve. Go to school the same old way. You know, stay in school. >> Stay in school. >> [applause] >> I guess I usually like to end with you're looking out on the crowd. There are a lot of people who I mean, I started this the opener with like I honestly look in the crowd and I see people who are not different than us per se, You know, we actually just are technical and like love systems.

嗯，而且你的系统性思维越强，越能编排上百万个智能体去自主解决问题，你就越有优势。所以系统性思维才会如此重要。但呃除此之外，我觉得这个世界还会继续给我们留下很多伟大的挑战去解决。就按老办法去上学吧。你知道的，好好待在学校里。>> 好好待在学校里。>> [掌声] >> 我想，嗯，我通常喜欢这样收尾，嗯，你在台上望着台下这群人。这里有很多人呃，我的意思是，我开场的时候就说过，我看着台下，我看到的这些人其实跟我们没什么本质不同，你知道的，我们其实就是技术型的人，喜欢琢磨系统。

系统思维的终极理由：未来的工作形态是『编排数百万个自主解决问题的智能体』。最后落点朴素：照常上学，好好读书。

orchestrate /ˈɔrkɪstreɪt/ *v.*
编排，统筹协调（原指配器、指挥乐队）

44:12 Garry Tan

How you know >> Thank you. Thank you. >> What advice would you give to this room of, you know, And you see yourself in this in this room and like I'm curious what you would say. If you could send a telegram, a message to the 18 to 22-year-old version of yourself, what would that be? >> I could tell you exactly how I felt when I first when Nvidia founded and the three of us started. The thing I felt at the time is there was so much for me to know and so much for me to learn. And I didn't know it. And I was telling you earlier there at the time there was there were no YouTube, there's you know, no YC, nobody's teaching you how to start a company. And so I went to the bookstore and I bought a book and the book said, "How to start a company?"

你怎么 >> 谢谢。谢谢。>> 你会给在座的各位什么建议，你知道，而且你在这个房间里能看到你自己的影子，我很好奇你会说什么。如果你能给 18 到 22 岁的自己发一封电报、一条消息，会是什么？ >> 我可以准确地告诉你，当英伟达刚创立、我们三个人刚起步时我的感受。嗯，我当时的感觉是，有太多东西是我需要知道的，太多东西是我需要学的。而我并不知道。我刚才跟你说过，那时候没有 YouTube，你知道，也没有 YC，没人教你怎么创业。所以我跑到书店买了一本书，那本书叫《如何创办一家公司》。

时代对照：1993 年没有 YouTube、没有 YC、没有创业教程，他买的《如何创办公司》厚达 500 页——自嘲读完公司早就倒了。Lori 是黄仁勋的妻子。

45:02 黄仁勋

Unfortunately, the book was like 500 pages long. And so I you know, I figured by the time I read it, you know, I'd be out of business. And Lor- Lori and I be out of money. And so there's no sense reading it. but the thing that the thing I remember very vividly is that how scared I was to go raise money because I felt that I was about to talk to a bunch of people and I didn't know how to answer their questions. And it's true. And I barely know how to answer their questions even today. but the thing that I learned is none of that stuff matters.

呃，可惜那本书大概有 500 页那么长。所以，你知道，我估摸着等我读完，公司早就倒闭了，我和 Lori 也早就没钱了。所以读它没什么意义。嗯，但我记得特别特别清楚的一件事是，我当时有多害怕去融资，因为我觉得我马上要面对一群人，而我不知道该怎么回答他们的问题。嗯，而且这是真的。哪怕到今天我也还是不太会回答他们的问题。呃但我学到的是，嗯，那些统统都不重要。

融资恐惧的自嘲：『至今也勉强能回答投资人的问题。』结论是那些让你恐惧的『不知道』其实都不重要——你永远会有不知道的东西。

vividly /ˈvɪvɪdli/ *adv.*
清晰生动地（remember vividly 记忆犹新）

45:41

As it turns out. And you're always going to have things that you don't know. And every single day the world's changing, technology changing. Obviously, this is the greatest time in the last 60 years to start a company. The whole industry has changed. It's a complete reset from a technology perspective. The single most important technology in human history, the computer, has been completely reset. And so, this is absolutely the single greatest time to start a company. And I'm jealous of all of you.

事实证明就是这样。而且你永远都会有你不懂的东西。每一天世界都在变，技术都在变。显然，现在是过去 60 年里创业的最好时机。整个行业都变了。从技术角度看，这是一次彻底的重置。人类历史上最重要的技术，计算机，被彻底重置了。所以，这绝对是创业最好的时机。而且我，我，我很嫉妒你们所有人。

> 明确判断：『这是 60 年来创业的最佳时机』。理由：人类历史上最重要的技术——计算机——正被彻底重置，整个技术栈从头重写，格局重新洗牌。

46:14

I and the opportunities you have ahead. I mean, it's going to be incredible. So, it's the perfect time on the one hand. On the other hand, the technology is changing so fast. And so, the question is, what's the right feeling for you? And eventually, and I told you the story of us of me buying the other book, the textbook. I think the psychology and the feeling that I have today on all of the new experiences and the new technology and new markets and new dynamics, I look at it and I say, this is important. I've got to go learn it.

我，还有你们面前的这些机会。我是说，那将会非常了不起。所以，一方面，这是最完美的时机。另一方面，技术变化得如此之快。所以问题是，对你来说什么才是对的感觉？最终，我跟你们讲过我买另一本书、那本教科书的故事。我今天面对所有这些新体验、新技术、新市场、新态势时的心态和感觉是，我看着它们然后说，这很重要。我得去学。

46:47

And I've got to go do something about it. And I better get to it as fast as I can. And how hard can it be? I always had this feeling, how hard can it be? And truth be told, it is way harder than you think. And but you don't want your mind to be to be there. You want your mind to be, how hard can it be? And let the suffering come to you a little bit at a time. You know, don't imagine how hard it's going to be and let all of that turn into anxiety and not doing something about it. You want to imagine your head, how hard can it be? You know, I've got a whole bunch of I've got a bunch of AI agents helping me anyways.

我得去做点什么。而且我最好尽快开始。还有，这能有多难？我一直都有这种感觉：这能有多难？而说实话，它比你想象的难得多。但是你不希望自己的脑子停在那儿。你希望自己的脑子想的是：这能有多难？然后让那些煎熬一点一点地找上你。你知道，别去想象它会有多难，别让那些全都变成焦虑，结果什么都不做。你要让脑子里想的是，这能有多难？你知道，反正我还有一大堆——我有一堆 AI 智能体在帮我呢。

『能有多难呢』的心理机制：不预先想象全部困难（那会变成焦虑和不行动），让苦难一点一点来。他承认真相是『远比想象难』，但心态要选择性乐观。

truth be told *phr.*
说实话，老实讲（插入语）

47:33

And so, how hard can it be? And then you get going on working on it. And so, that's probably the attitude of an entrepreneur. You know you have to learn a bunch of stuff along the way. You believe in your ability to learn. Which is, you know, learning is the single greatest superpower. And if you go into it with the attitude, how hard can it be? If anybody can do it, I can do it. And just realize that it will be hard and you just have to have the resilience to overcome it every single day. You don't have to overcome life in one day. You just have to overcome that morning. That morning, you know, you have to overcome today. And so it's not a big deal. Just get through today. Wait till right? Work towards tomorrow. Keep following your dreams.

所以，这能有多难？然后你就开始动手干。所以，这大概就是一个创业者的心态。你，你知道，你必须一路上学一堆东西。你要相信自己的学习能力。要知道，学习是唯一最强大的超能力。如果你抱着这样的心态去做：这能有多难？如果别人能做到，我也能做到。同时也要明白它确实会很难，你只需要有那份韧性，每一天都去克服它。你不需要在一天之内战胜整个人生。你只需要战胜那个早晨。那个早晨，你知道，你只要战胜今天的今天。所以这没什么大不了。撑过今天就好。等着，对吧？为明天努力。继续追随你的梦想。

韧性的可操作定义：不必一天战胜整个人生，只需撑过今天早晨。把长期困难切成以『天』为粒度——全场最具迁移价值的心态建议之一。

resilience /rɪˈzɪliəns/ *n.*
韧性，抗逆力

48:17

And the rest of everything if you stick if you stick with it long enough, you know, Nvidia happens. And so, you know, I think that the wisdom that I can if it there's anything is resilience is probably the single most important thing. And if you believe in something, just get going on it and get your mind you know, out of out of keeping your yourself from pursuing it because of you know, fear or anxiety or lack of confidence or whatever it is. And then you're just going to tell yourself I'm going to learn my way there.

剩下的一切，只要你坚持得够久，呃，你知道的，英伟达就这么诞生了。所以，你知道，我觉得如果说我能给出什么智慧的话，韧性大概是唯一最重要的东西。如果你相信某件事，那就马上动手去做，别让自己的脑子，你知道，因为恐惧、焦虑、不自信或者别的什么，而阻止你去追求它。然后你就告诉自己：我会一边学一边走到那里。

收束：韧性是他能给出的最重要的一条智慧。恐惧、焦虑、缺乏自信都不该阻止行动——『我会一路学着走到那里』呼应开头买教科书自学的故事。

stick with it *phr.*
坚持下去，不放弃

48:50 Garry Tan

>> Jensen Huang everybody. >> All right, guys. Thank you. >> Thank you so much. Yes, it was >> Thank you guys.

>> 大家掌声送给黄仁勋。>> 好了，各位。谢谢。>> 非常感谢。是的，这真是 >> 谢谢大家。

第二部分　生词总表 · 复习用

词汇	音标	词性	释义	出现
a thousand flowers bloom	—	phr.	百花齐放（let a thousand flowers bloom，允许多种做法并存）	25:31
abstractly	/æbˈstræktli/	adv.	抽象地（think abstractly 抽象思考）	20:38
accelerators	/ækˈsɛləreɪtərz/	n.	加速器（辅助 CPU 的专用计算硬件）	1:31
across the board	—	phr.	全面地，各领域无一例外地	33:23
agentically	/eɪˈdʒɛntɪkli/	adv.	以（AI）智能体方式地（AI 行业新词）	19:50
alpha	/ˈælfə/	n.	（金融）超额收益；（喻）人无我有的优势	29:40
around the corner	—	phr.	即将到来，近在眼前	35:10
articulation	/ɑrˌtɪkjəˈleɪʃən/	n.	关节活动；（机器人）关节运动控制	34:00
asynchronously	/eɪˈsɪŋkrənəsli/	adv.	异步地（不阻塞主流程地后台进行）	21:44
at your fingertip	—	phr.	触手可及（常作 at one's fingertips）	30:51
augment	/ɔːgˈmɛnt/	v.	增强，扩充（在原有基础上加强）	1:31
autonomous	/ɔˈtɑnəməs/	adj.	自主的，自治的	25:01
backlog	/ˈbæklɔg/	n.	积压（的工作、订单、病例）	32:37
be of service to	—	phr.	为……效力，对……有用	14:18
bodes well for	—	phr.	对……是好兆头（bode well/ill for）	30:18
bottlenecks	/ˈbɑtlnɛks/	n.	瓶颈	24:24
causality	/kɔˈzæləti/	n.	因果关系	35:10
chain of thought	—	phr.	思维链（让模型分步推理的技术）	25:31
coarse	/kɔrs/	adj.	粗糙的；粗粒度的（与 fine-grained 相对）	20:38
cognitive	/ˈkɑgnətɪv/	adj.	认知的（cognitive tasks 认知类任务）	30:51
concurrency	/kənˈkɜrənsi/	n.	（计算）并发，并发性	24:24
confront	/kənˈfrʌnt/	v.	直面，正视（confront the fact/reality 直面事实）	2:47
conjectured	/kənˈdʒɛktʃərd/	v.	推测，猜想（conjecture）	37:24
decompose	/ˌdikəmˈpoʊz/	v.	分解，拆解（新情境拆成已知要素）	26:50
delta	/ˈdɛltə/	n.	增量，差值（工程用语，makes a delta difference）	22:18

domain specific	—	adj./phr.	领域专用的（domain-specific AI 领域专用 AI）	29:07
eating glass	—	phr.	吞玻璃（喻创业的极端艰辛，源自马斯克名言）	7:06
empower	/ɪmˈpaʊər/	v.	赋能，使有能力	15:13
eval	/ɪˈvæl/	n.	评测（evaluation 的 AI 行业缩略语）	36:32
existential	/ˌɛgzɪˈstɛnʃəl/	adj.	关乎存亡的；存在层面的	22:57
fine-grained	/ˈfaɪn greɪnd/	adj.	细粒度的，精细的	21:44
fine-tune	/ˌfaɪnˈtun/	v.	微调（在已训练模型上做针对性再训练）	36:01
first principles	—	phr.	第一性原理（从最基本事实出发推理）	12:27
flawed	/flɔːd/	adj.	有缺陷的（fundamentally flawed 存在根本缺陷）	7:38
flippant	/ˈflɪpənt/	adj.	轻率的，不当回事的	29:40
flywheel	/ˈflaɪwil/	n.	飞轮；（喻）自我强化的增长循环	37:24
foundational	/faʊnˈdeɪʃənl/	adj.	基础性的，根本的	12:27
frontier	/frʌnˈtɪr/	adj./n.	前沿（的）；边界	19:02
fulfill	/fʊlˈfɪl/	v.	履行（合同、承诺）；实现	7:38
game changing	—	adj./phr.	颠覆格局的，革命性的	22:57
harness	/ˈhɑrnɪs/	n.	挽具；（软件）执行框架、运行环境	27:54
imprecise	/ˌɪmprɪˈsaɪs/	adj.	不精确的	11:50
inclination	/ˌɪnkləˈneɪʃən/	n.	倾向，意愿（first inclination 第一反应）	14:18
intersecting	/ˌɪntərˈsɛktɪŋ/	adj.	交叉的，相交的（intersecting domains 交叉领域）	41:35
intimate	/ˈɪntəmət/	adj.	熟稔的，有深入了解的（be intimate about sth）	23:47
into the weeds	—	phr.	深入琐碎细节（go all the way into the weeds）	13:30
introverted	/ˈɪntrəvɜrtɪd/	adj.	内向的	39:10
lens	/lɛnz/	n.	透镜；（喻）看问题的视角（our lens of the world）	10:21
long division	—	phr.	长除法（笔算多位数除法）	41:35
middleware	/ˈmɪdlwɛr/	n.	中间件（连接系统与应用的软件层）	12:27
molecular dynamics	—	phr.	分子动力学（模拟分子运动的计算方法）	5:29
move the cheese	—	phr.	动奶酪（喻打破现状令人不安，出自《谁动了我的奶酪》）	30:18
north of	—	phr.	超过，多于（口语，用于数字前）	9:37

off the shelf	—	phr.	现成的，即买即用的	29:40
on tap	—	phr.	（像自来水般）随取随用的	30:18
open weights	—	phr.	开放权重（公开模型参数供下载使用）	39:47
open-sourced	/ˌoʊpən ˈsɔrst/	v.	将……开源（公开源代码）	38:07
orchestrate	/ˈɔrkɪstreɪt/	v.	编排，统筹协调（原指配器、指挥乐队）	43:23
outsource	/ˈaʊtsɔrs/	v.	外包	22:57
paralegal	/ˌpærəˈligl/	n.	律师助理	33:23
per se	/ˌpɜr ˈseɪ/	adv.	本身，就其本身而言（拉丁语借词）	9:37
pipelines	/ˈpaɪplaɪnz/	n.	管线；（图形学）渲染流水线	3:25
pivotal	/ˈpɪvətl/	adj.	关键性的，起决定作用的	7:06
prior knowledge	—	phr.	先验知识（事先已具备的知识）	26:14
radiology	/ˌreɪdiˈɑlədʒi/	n.	放射学；放射科（读 X 光、CT 等影像）	32:37
ramp it up	—	phr.	爬坡提升；逐步扩大产量（ramp up）	24:24
recursive self-improvement	—	phr.	递归自我改进（AI 通过改进自身而加速变强）	20:38
reinforcement learning	—	phr.	强化学习（通过奖励信号试错学习的方法）	36:01
resilience	/rɪˈzɪliəns/	n.	韧性，抗逆力	47:33
ruffles feathers	—	phr.	惹恼人，得罪人（ruffle sb's feathers）	13:30
sandboxes	/ˈsændbɑksɪz/	n.	沙箱（隔离的安全运行环境）	24:24
scalable	/ˈskeɪləbl/	adj.	可扩展的，可规模化的	26:14
spot-on	/ˌspɑːt ˈɑːn/	adj.	完全正确的，精准的（口语）	5:29
stay on top of	—	phr.	掌控，驾驭（局面）；字面为稳站在……之上	15:46
stick with it	—	phr.	坚持下去，不放弃	48:17
surreal	/səˈriəl/	adj.	超现实的；恍如梦境般不真实的	0:34
switch gears	—	phr.	换挡；（喻）转换话题或方向	19:02
synthesizing	/ˈsɪnθəsaɪzɪŋ/	v.	综合；（芯片设计）自动综合生成电路	19:50
tactile	/ˈtæktaɪl/	adj.	触觉的（tactile sensation 切身的触感）	15:46
takeaway	/ˈteɪkəweɪ/	n.	核心收获，要点（the big takeaway 最大心得）	41:01
traits	/treɪts/	n.	特质，品性	16:45
truth be told	—	phr.	说实话，老实讲（插入语）	46:47

tweaking	/ˈtwikɪŋ/	v.	微调，小幅调整（tweak）	18:32
uneven	/ʌnˈivən/	adj.	不均衡的，不平均的	30:51
universal function approximator	—	phr.	通用函数逼近器（神经网络可拟合任意函数的理论性质）	11:00
valuation	/ˌvæljuˈeɪʃən/	n.	估值	9:37
via	/ˈvaɪə/	n.	（电路板、芯片中的）过孔；此处非介词用法	22:18
vividly	/ˈvɪvɪdli/	adv.	清晰生动地（remember vividly 记忆犹新）	45:02
whatever it takes	—	phr.	不惜一切代价，无论付出什么	17:59