# Andrej Karpathy在各个场合的观点

2025年6月19日 2 分钟阅读

#AI #Andrej Karpathy

> *Andrej Karpathy在各个场合的观点*

## Andrej Karpathy: 软件正在改变 (Again)

**会议:** 安德烈·卡帕西在旧金山举行的 AI 创业学校上的主题演讲。
**日期:** 2025年6月19日 **主题:** AI时代的软件范式转变

### I. 核心观点：软件范式演进

Karpathy指出，软件正在经历70年来最根本的变革，并在过去几年中快速经历了两次重大转变。

**软件1.0 (Software 1.0):**

**定义:** 传统的人类编写的显式代码 (如C++代码) [01:42]。

**特点:** 指令明确，逻辑硬编码。

**软件2.0 (Software 2.0):**

**定义:** 神经网络及其权重 [01:48]。

**编程方式:** 不再直接编写代码，而是通过调整数据集和优化器来生成神经网络参数。

**生态系统:** Hugging Face被视为软件2.0领域的GitHub [02:12]。

**演进观察 (特斯拉自动驾驶案例):** 在特斯拉自动驾驶项目中，随着神经网络（软件2.0）能力的增强，大量的C++代码（软件1.0）被删除，功能逐渐迁移到神经网络中，软件2.0“吞噬”了软件1.0堆栈 [05:01]。

**软件3.0 (Software 3.0):**

**定义:** 可通过自然语言（如英语）提示进行编程的大型语言模型 (LLMs) [03:06, 03:19]。

目录

文章信息

字数

阅读时间

发布时间

更新时间

标签

#AI #Andrej Karpathy

**编程语言:** 英语成为一种非常有趣的编程语言。

**范式转变:** 带来了全新的编程范式，代码中开始出现大量自然语言与代码的混合。

## II. LLMs作为新型计算范式与生态系统

Karpathy将LLMs视为一种新型的计算机或操作系统，并进行了多维度类比。

**LLMs的"公用事业"属性 (Utilities):**

**投资模式:** LLM实验室（如OpenAI, Gemini, Anthropic）投入大量资本（Capex）进行模型训练，相当于建设电网；运营成本（Opex）通过API提供智能服务，按token计费 [06:35, 06:50]。

**需求特点:** 用户对API有低延迟、高可用、一致性质量等类似公用事业的需求。

**竞争与故障:** LLMs作为软件不争夺物理空间，允许多个提供商并存。当领先的LLMs宕机时，他形容为全球范围内的"智能限电"（intelligence brownout），表明我们对这些模型的依赖性日益增长 [07:42]。

**LLMs的"晶圆厂"属性 (Fabs):**

**高资本投入:** 训练LLMs需要巨大的资本投入 [08:00]。

**技术树与研发:** 核心技术和研发秘密集中在少数LLM实验室内部。

**类比:** 使用Nvidia GPU训练模型类似"无晶圆厂"模式；谷歌自建TPU并训练模型则类似"英特尔"模式，拥有自己的"晶圆厂"。

**LLMs的"操作系统"属性 (Operating Systems - 最强类比):**

**复杂生态系统:** LLMs不再是简单的商品，而是日益复杂的软件生态系统 [09:12]。

**市场格局:** 类似于Windows/macOS (闭源) 和Linux (开源) 的模式，LLM领域也存在少数闭源提供商和类似Llama生态系统的开源替代方案 [09:32]。

**内部结构类比:** LLM本身类似CPU，上下文窗口（context window）类似内存，LLM负责协调内存和计算以解决问题 [10:16]。

**应用部署:** LLM应用（如Cursor）可以在不同LLM（如GPT、Claude、Gemini）上运行，类似于应用可以在Windows、Linux、Mac上运行 [10:53]。

**1960年代计算时代:** 当前LLM计算昂贵且集中于云端，通过分时（time-sharing）访问 [11:02]。个人计算革命尚未到来，但Mac

mini等设备可能预示着早期迹象 [11:39]。

**GUI缺失:** 目前与LLM的交互（如ChatGPT）更像是通过终端与操作系统对话，通用的图形用户界面（GUI）尚未被发明 [12:24]。

**技术扩散方向的"反转" (Flipped Diffusion):**

**传统模式:** 历史上，变革性技术（如电力、计算机、互联网）通常首先由政府和大型企业采用，随后才扩散到消费者。

**LLMs模式:** LLMs的扩散方向是"反向"的，它们首先在消费者手中普及（例如用于煮鸡蛋等日常任务），而政府和企业在采用上反而滞后 [12:57, 13:18]。

## III. LLMs的"心理学"与局限性

Karpathy将LLMs比作"人的精神"（people spirits）或"人的随机模拟"（stochastic simulations of people），并分析了它们的特点和认知缺陷。

**超能力:**

**百科全书式知识与记忆:** 拥有远超个体人类的知识和记忆，能记住大量信息，类似电影《雨人》中的自闭症学者 [15:30, 15:41]。

**认知缺陷:**

**幻觉 (Hallucinations):** 经常虚构信息。

**锯齿状智能 (Jagged Intelligence):** 在某些问题解决领域表现超人，但在其他领域会犯人类不会犯的基本错误（如9.11大于9.9，或strawberry有两个R） [16:20]。

**顺行性遗忘症 (Anterograde Amnesia):** LLMs不会像人类一样通过睡觉巩固知识或随时间积累上下文和专业知识。上下文窗口是其"工作记忆"，需要直接编程 [16:43]。他推荐电影《记忆碎片》和《50次初恋》来理解这种特性 [17:15]。

**易受攻击性:** 容易受到提示注入（prompt injection）攻击，可能泄露数据 [17:44]。

## IV. LLM时代的机会与挑战

Karpathy探讨了如何利用LLMs的超能力并规避其缺陷，以及未来的发展方向。

**部分自主应用 (Partial Autonomy Apps):**

**趋势:** 许多软件将变得部分自主，而不是完全自主 [21:30]。

**理想LLM应用的属性:**

**上下文管理:** LLMs处理大量上下文管理 [19:24]。

**多LLM协调:** 协调对多个LLM的调用（如Cursor在后台调用嵌入模型、聊天模型、代码diff模型） [19:33]。

**应用特定GUI:** GUI对于人类审计LLM的输出至关重要，比纯文本交互更高效、直观 [19:50, 22:34]。

**自主性滑块 (Autonomy Slider):** 用户可以根据任务复杂性调整AI的自主程度，从辅助完成到完全代理模式 [20:21, 28:13]。Cursor和Perplexity是良好范例 [20:32, 21:17]。

**人机协作与“牵制”AI (Keeping AI on the Leash):**

**核心目标:** 加速人类验证与AI生成之间的循环（generation-verification loop） [22:10]。

**加速验证:** GUI通过利用人类的视觉处理能力，使得审核AI生成的内容更快、更愉快 [22:34]。

**限制AI的自主性:** AI代理不应被赋予过高自主权，因为生成过大的代码修改（如10,000行diff）会成为人类审核的瓶颈 [22:53]。

**具体提示 (Concrete Prompts):** 编写更具体、清晰的提示能提高AI输出的准确性，减少验证失败和反复修改 [24:41]。

**教育应用案例:** 在教育领域，通过设计可审核的课程内容和明确的教学大纲，可以有效“牵制”AI，防止其“迷失在森林中” [25:05]。

**“钢铁侠战衣”类比 (Iron Man Suit Analogy):**

**增强与代理:** 理想的AI工具应兼具人类增强（augmentation）和自主代理（agent）的特性 [27:54]。

**当前重点:** 鉴于LLMs的局限性，目前应更多地构建“钢铁侠战衣”（部分自主产品，强调UI/UX和快速验证循环），而非完全自主的“钢铁侠机器人” [28:22, 28:34]。

**未来展望:** 未来的十年，自主性滑块将逐渐向右移动，产品会变得更加自主 [39:08]。

**自然语言编程与“Vibe Coding”:**

**全民程序员时代:** 英语作为自然接口，使得人人都能成为程序员，这是前所未有的巨大机遇 [29:11, 29:17]。

**Vibe Coding:** 一种通过自然语言快速构建自定义应用的方式，即使是非专业人士也能实现（例如Karpathy用Vibe Coding开发了iOS应用和Menu Genen） [30:32, 31:07]。

**挑战:** 实际部署（认证、支付、域名、部署）比编写核心代码更耗时，因为这些DevOps任务目前仍需人工点击GUI完成 [32:21]。

**为AI代理构建 (Building for Agents):**

**新消费者:** AI代理是数字信息的新型消费者和操作者，现有软件基础设施需为此进行调整 [33:48, 34:02]。

`lm.txt`: 类似于`robots.txt`，用于指导LLMs如何与网站交互的简单Markdown文件 [34:18]。

**LLM友好文档:** 将文档转换为Markdown格式，并将"点击"等人类操作指令替换为LLM可执行的命令（如curl命令），如Vercel和Stripe已在实践 [34:43, 36:04]。

**数据摄取工具

## V. 总结与展望

**黄金时代:** 当前是进入软件行业的绝佳时机，大量代码需要重写，专业人士和"Vibe Coder"都将发挥作用 [38:16]。

**LLMs的本质:** LLMs兼具公用事业、晶圆厂和操作系统的特性，但仍处于早期阶段（类似于1960年代的操作系统） [38:25]。

**与"有缺陷的人类精神"共事:** 我们必须学会与这些拥有超能力但存在认知缺陷的"人类精神"合作 [38:42]。

**调整基础设施:** 需要调整基础设施以适应LLMs，构建部分自主产品，并通过工具加速人机协作循环 [38:48]。

**未来十年：自主性提升:** 在未来十年，自主性滑块将持续向右移动，产品将变得更加自主。

## 参考


Youtube视频：软件正在改变 (Again)

2017 年的软件 2.0 博客文章

大型语言模型如何颠覆技术扩散的剧本

Vibe 编码 MenuGen（回顾）

分享这篇文章

## 相关文章推荐

### Ilya Sutskever...

Ilya Sutskever 在各个场合的观点

### SkyworkAI DeepResearc

SkyworkAI DeepResearchAge

### Google I/O 2025 大会...

本文介绍了 Google I/O 20...