Lecture

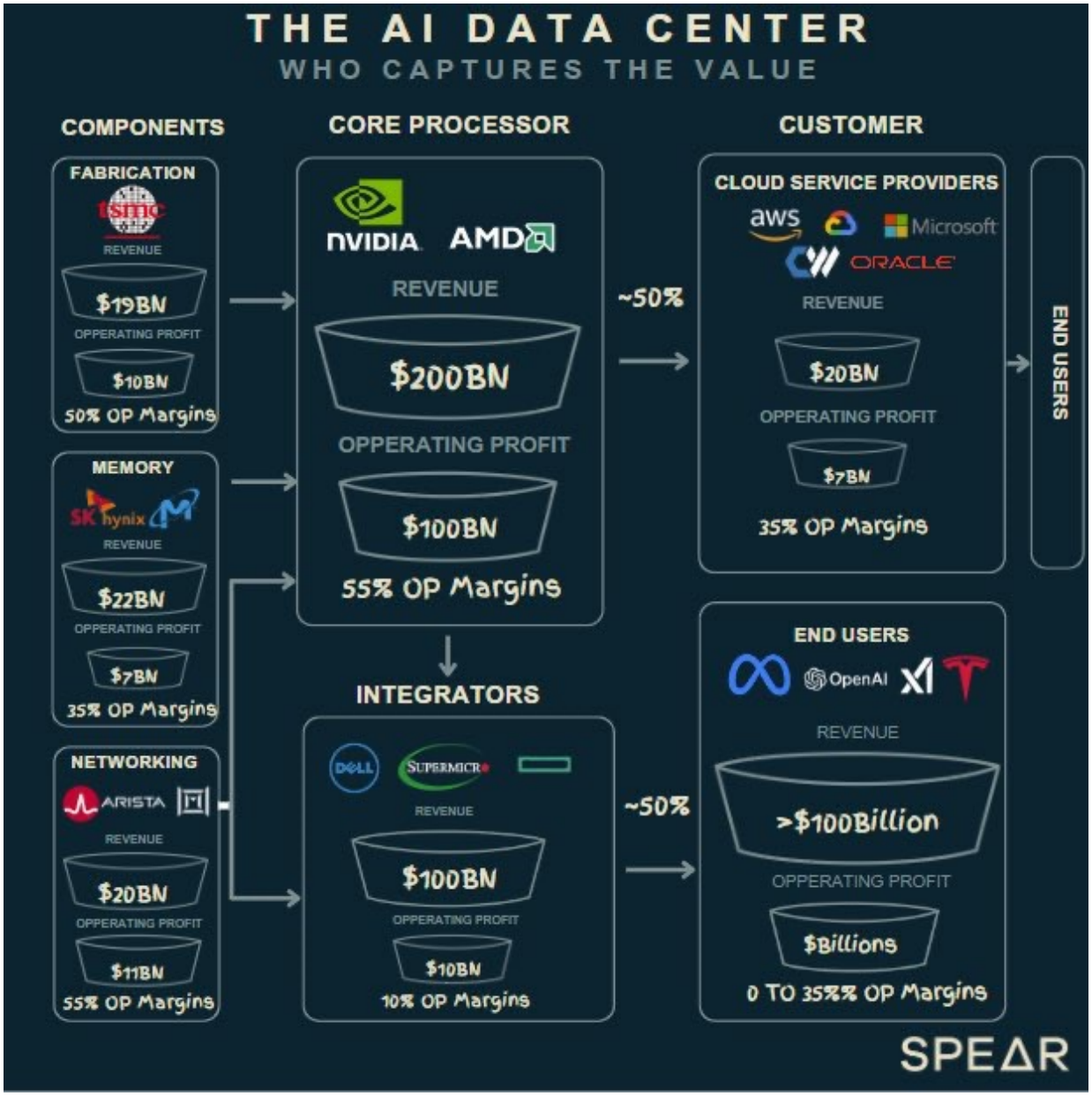
AI Trends

What are We Talking About Today?

- **AI Product Landscape**
  - Nvidia rules the world
  - ~80% of AI-native companies are building agentic workflows
  - Vertical AI apps (industry-specific) and horizontal apps remain widespread
- **Model & Infrastructure Choices**
  - Cloud-based APIs (e.g., OpenAI, Anthropic, Gemini) are preferred over on-prem
  - Flexibility drives model-swapping architectures and selective use of open source
- **Adoption & Productivity**
  - ~70% of employees have access to AI tools, but only ~50% use them actively
  - Coding assistance is the highest-impact use case for productivity gains
  - Orchestration frameworks (LangChain, Hugging Face) and safety layers (Guardrails) are gaining traction
  - Vibe coding and Software 3.0 the new paradigm shift

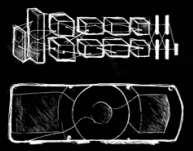
What are We Talking About Today? (cont'd)

- **What's New**
  - Top AI Stories of 2025 – Deeplearning.ai
  - 2025 LLM Year in Review– Andrej Karpathy
  - AI Datacenters in Space – Starcloud and SpaceX
  - Nvidia's $20B Groq "Acquisition"
  - Karpathy's Inspired Coding Guidelines


THE AI DATA CENTER
WHO CAPTURES THE VALUE
COMPONENTS
CORE PROCESSOR
CUSTOMER
FABRICATION
REVENUE
$19BN
OPPERATING PROFIT
$10BN
50% OP Margins
MEMORY
REVENUE
$22BN
OPPERATING PROFIT
$7BN
35% OP Margins
NETWORKING
REVENUE
$20BN
OPPERATING PROFIT
$11BN
55% OP Margins
NVIDIA
AMD
REVENUE
$200BN
OPPERATING PROFIT
$100BN
55% OP Margins
~50%
INTEGRATORS
REVENUE
$100BN
OPPERATING PROFIT
$10BN
10% OP Margins
~50%
CLOUD SERVICE PROVIDERS
aws
Microsoft
ORACLE
REVENUE
$20BN
OPPERATING PROFIT
$7BN
35% OP Margins
END USERS
END USERS
OpenAI
REVENUE
>$100Billion
OPPERATING PROFIT
$Billions
0 TO 35% OP Margins
SPEAR


NVIDIA

2012 ALEXNET

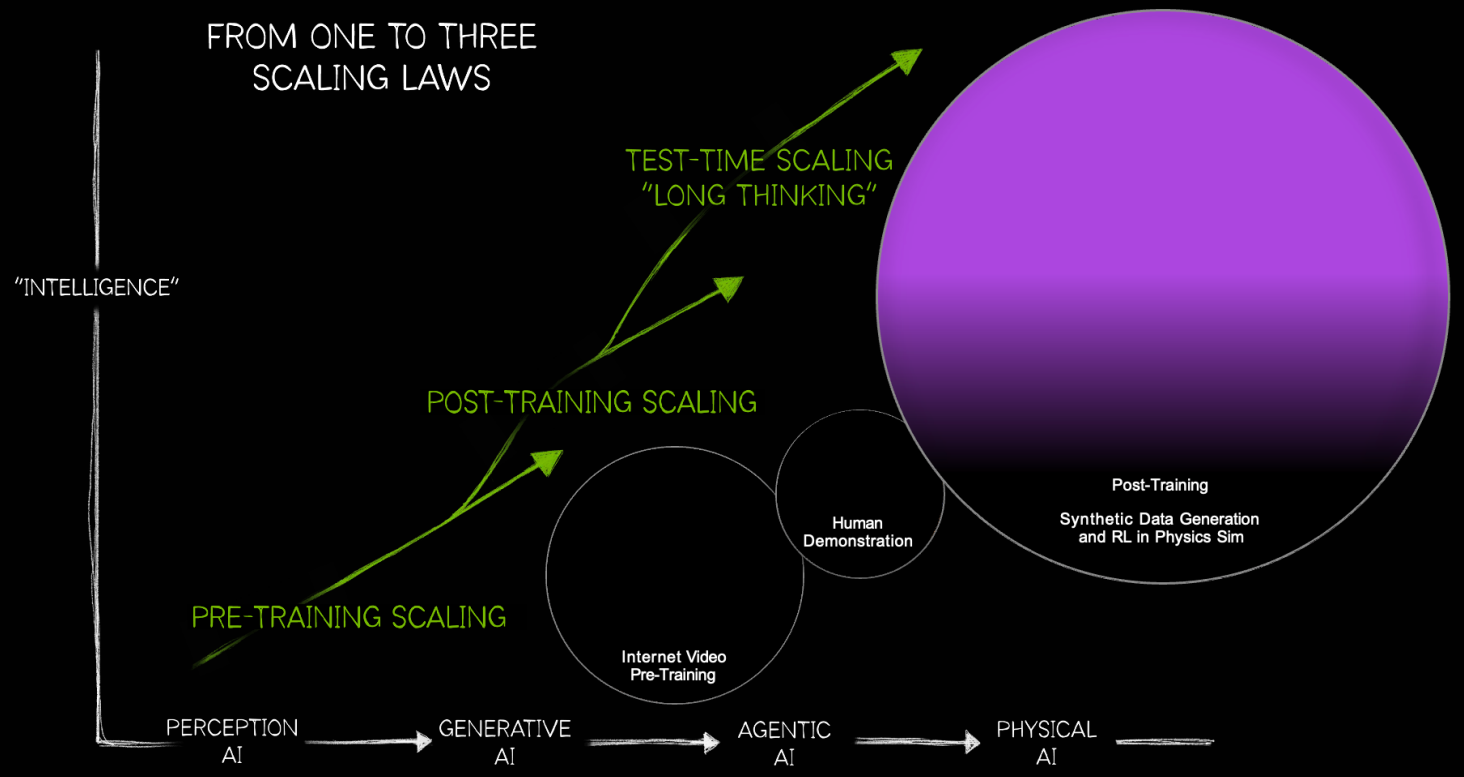


PERCEPTION AI
SPEECH RECOGNITION
DEEP RECSYS
MEDICAL IMAGING

GENERATIVE AI
DIGITAL MARKETING
CONTENT CREATION

AGENTIC AI
CODING ASSISTANT
CUSTOMER SERVICE
PATIENT CARE

PHYSICAL AI
AUTONOMOUS VEHICLES
GENERAL ROBOTICS


FROM ONE TO THREE
SCALING LAWS
"INTELLIGENCE"
TEST-TIME SCALING
"LONG THINKING"
POST-TRAINING SCALING
PRE-TRAINING SCALING
Post-Training
Synthetic Data Generation
and RL in Physics Sim
Human
Demonstration
Internet Video
Pre-Training
PERCEPTION
AI
GENERATIVE
AI
AGENTIC
AI
PHYSICAL
AI

CUDA-X FOR EVERY INDUSTRY

WARP
PHYSICS

cuDF
cuML
DATA SCIENCE
AND PROCESSING

cuDSS
cuSPARSE
cuFFT
AMGX
COMPUTER AIDED
ENGINEERING

TRT-LLM
MEGATRON
NCCL
cuDNN
CUTLASS
cuBLAS
DEEP
LEARNING

cuEQUIVARIANCE
cuTENSOR
QUANTUM
CHEMISTRY

cuQUANTUM
CUDA-Q
QUANTUM
COMPUTING

EARTH-2
WEATHER
ANALYTICS

MONAI
MEDICAL
IMAGING

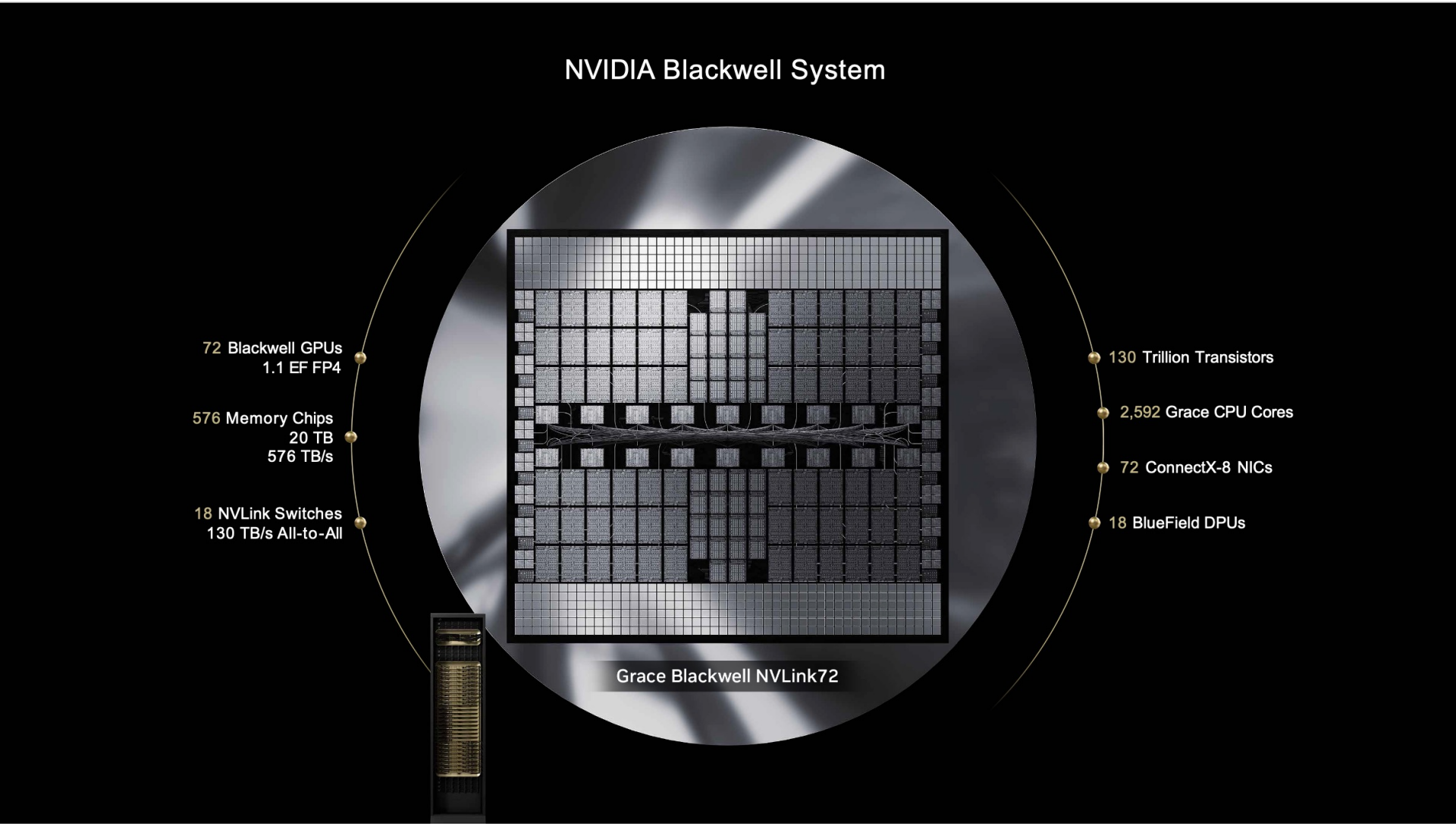
PARABRICKS
GENE
SEQUENCING

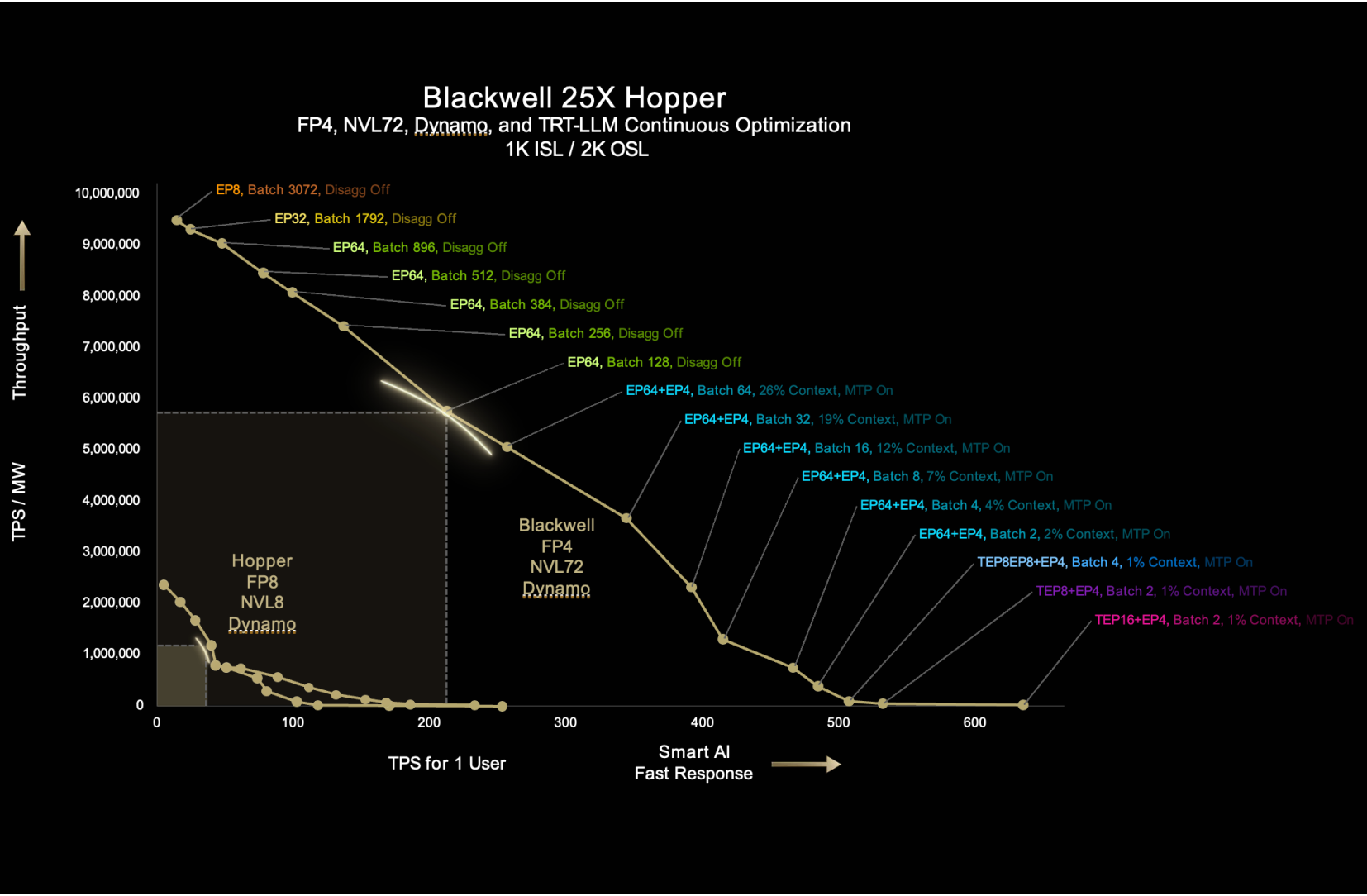
cuOPT
DECISION
OPTIMIZATION

AERIAL
SIONNA
5G/6G
SIGNAL
PROCESSING

cuLITHO
COMPUTATIONAL
LITHOGRAPHY

cuPYNUMERIC
NUMERICAL
COMPUTING


NVIDIA Blackwell System
72 Blackwell GPUs
1.1 EF FP4
576 Memory Chips
20 TB
576 TB/s
18 NVLink Switches
130 TB/s All-to-All
130 Trillion Transistors
2,592 Grace CPU Cores
72 ConnectX-8 NICs
18 BlueField DPUs
Grace Blackwell NVLink72


Blackwell 25X Hopper
FP4, NVL72, Dynamo, and TRT-LLM Continuous Optimization
1K ISL / 2K OSL
TPS / MW
Throughput
10,000,000
9,000,000
8,000,000
7,000,000
6,000,000
5,000,000
4,000,000
3,000,000
2,000,000
1,000,000
0
0
100
200
300
400
500
600
TPS for 1 User
Smart AI
Fast Response
EP8, Batch 3072, Disagg Off
EP32, Batch 1792, Disagg Off
EP64, Batch 896, Disagg Off
EP64, Batch 512, Disagg Off
EP64, Batch 384, Disagg Off
EP64, Batch 256, Disagg Off
EP64, Batch 128, Disagg Off
EP64+EP4, Batch 64, 26% Context, MTP On
EP64+EP4, Batch 32, 19% Context, MTP On
EP64+EP4, Batch 16, 12% Context, MTP On
EP64+EP4, Batch 8, 7% Context, MTP On
EP64+EP4, Batch 4, 4% Context, MTP On
EP64+EP4, Batch 2, 2% Context, MTP On
TEP8EP8+EP4, Batch 4, 1% Context, MTP On
TEP8+EP4, Batch 2, 1% Context, MTP On
TEP16+EP4, Batch 2, 1% Context, MTP On
Hopper
FP8
NVL8
Dynamo
Blackwell
FP4
NVL72
Dynamo

Oberon Rack
Liquid Cooled

Blackwell Ultra NVL72

Second Half 2025

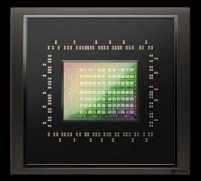
1.1 EF Dense FP4 Inference
0.36 EF FP8 Training
1.5X GB200 NVL72

New Attention Instructions
2X

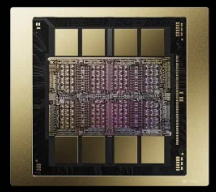
20 TB HBM | 40 TB Fast Memory
1.5X

14.4 TB/s CX8
2X

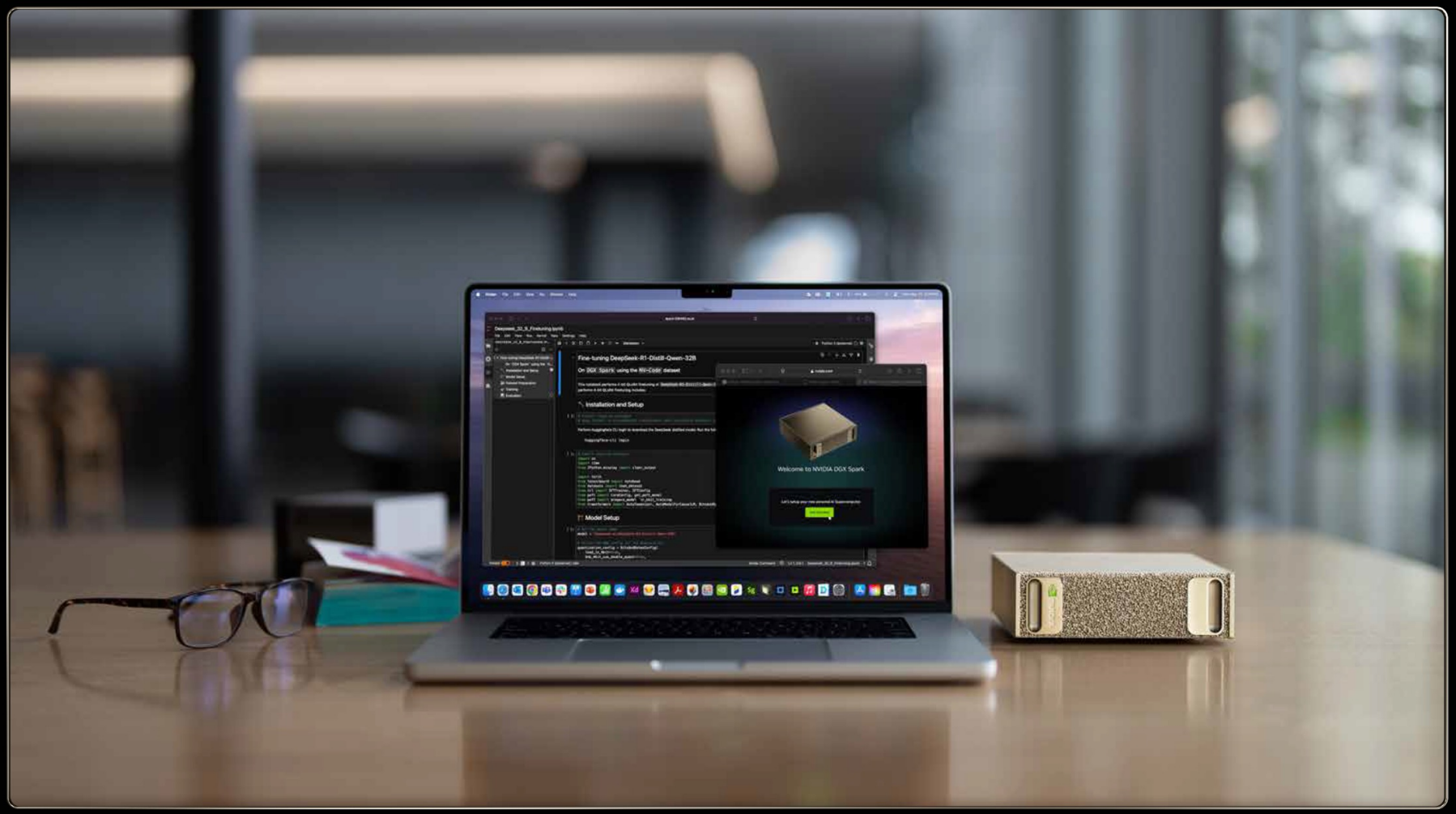
Grace

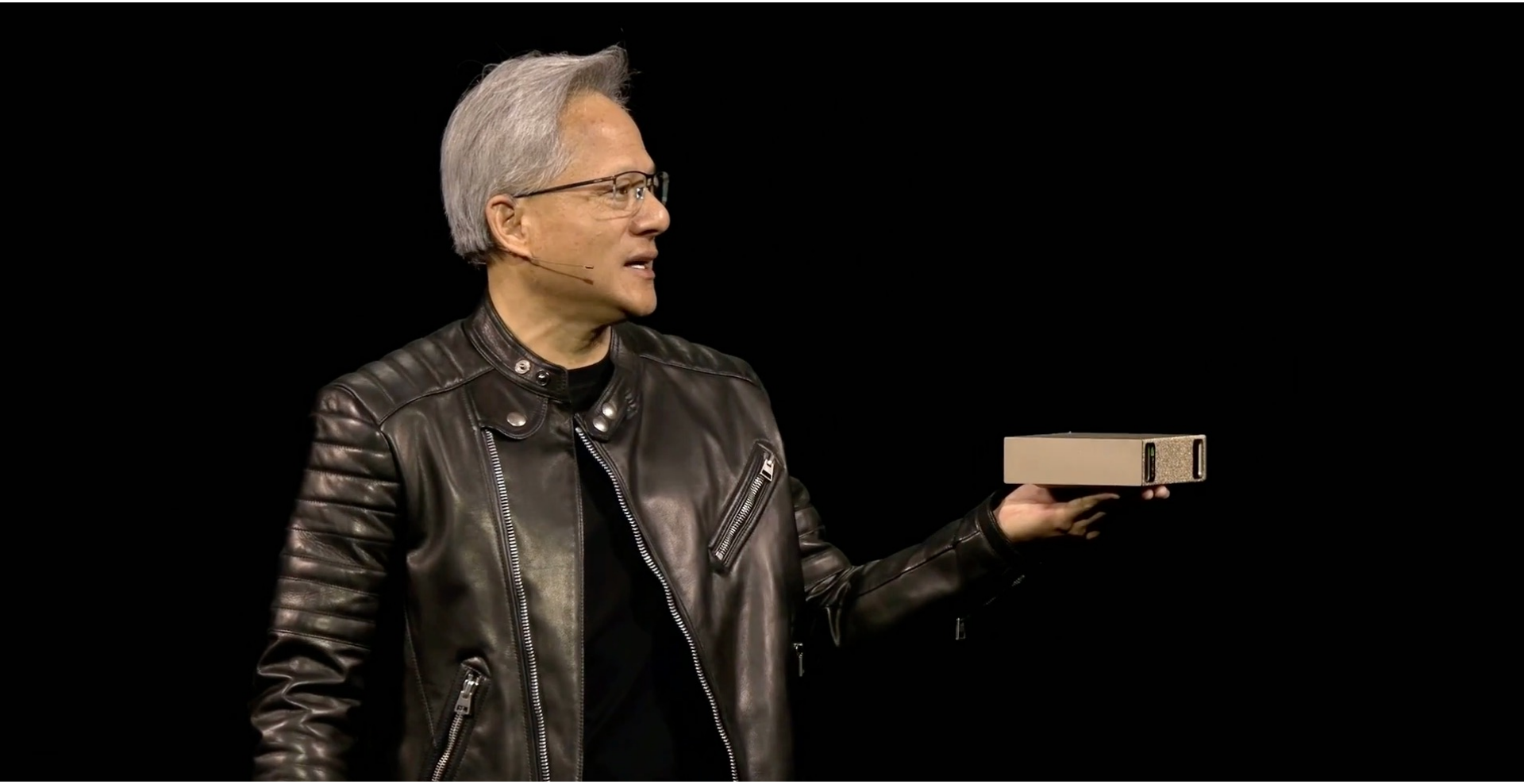


Blackwell Ultra



2 Reticle-Sized GPUs
15PF Dense FP4 | 288GB HBM3e


Fine-tuning DeepSeek-R1-Distill-Qwen-32B
On DGX Spark using the NV-Code dataset
Installation and Setup
Model Setup
Welcome to NVIDIA DGX Spark



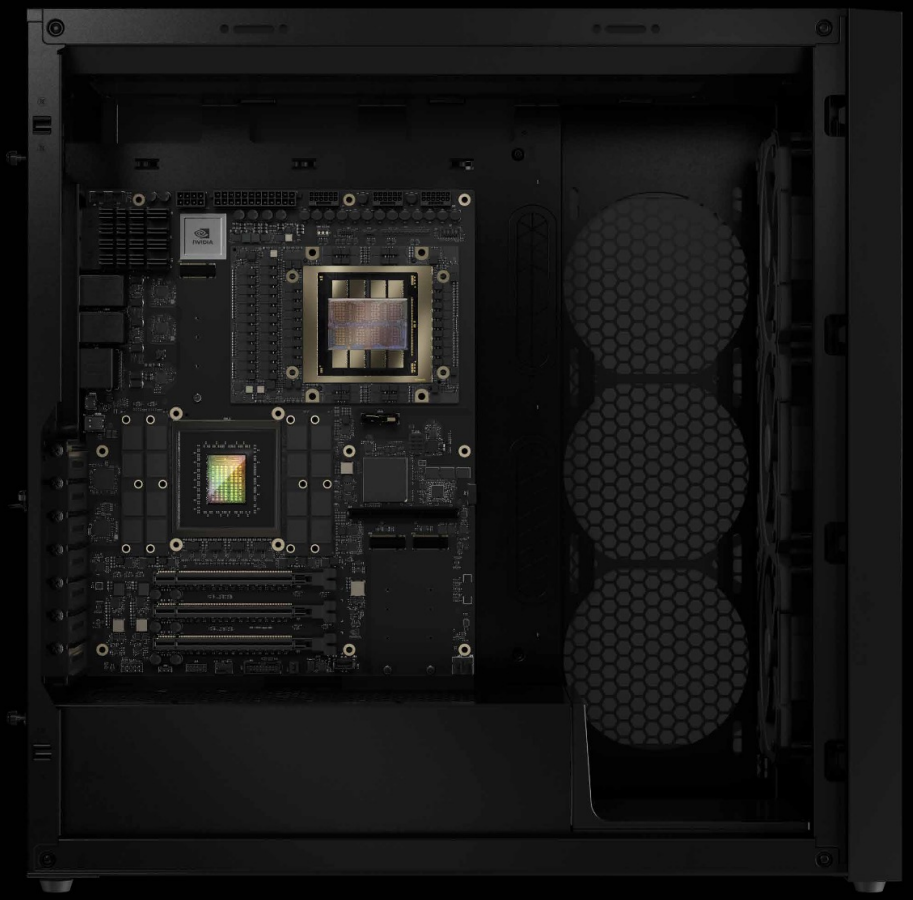
GTC 2025:
Introducing DGX Spark

- Part of a new class of computers created for the AI era
- Designed from the ground up to build and run AI
- Full NVIDIA accelerated AI software stack
- DGX OS
- Available from OEM partners
- Reserve systems now at NVIDIA.com Marketplace

ASUS DELL Technologies hp Lenovo

https://www.nvidia.com/en-us/on-demand/session/gtc25-s74680/


1009126
TESLA
DINER
HOLLYWOOD, CA
PLW3A



DGX Station

The Ultimate Workstation
for AI and Data Science

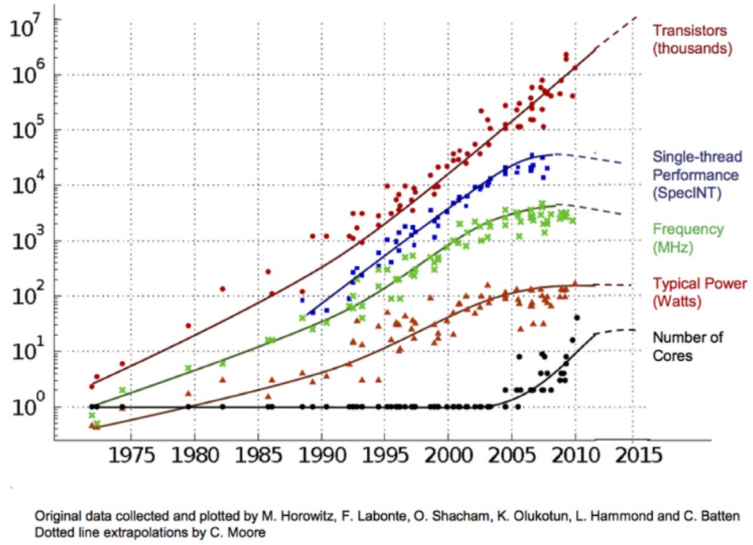
- GB300 Superchip
- 784 GB Unified System Memory
- 20,000 AI TFLOPS
- ConnectX-8 SuperNIC

ASUS BOXX DELL Technologies hp Lambda SUPERMICRO

The end of **Dennard Scaling**

The end of Dennard scaling means that power grows with transistor count and corresponding frequency improvements are no longer possible

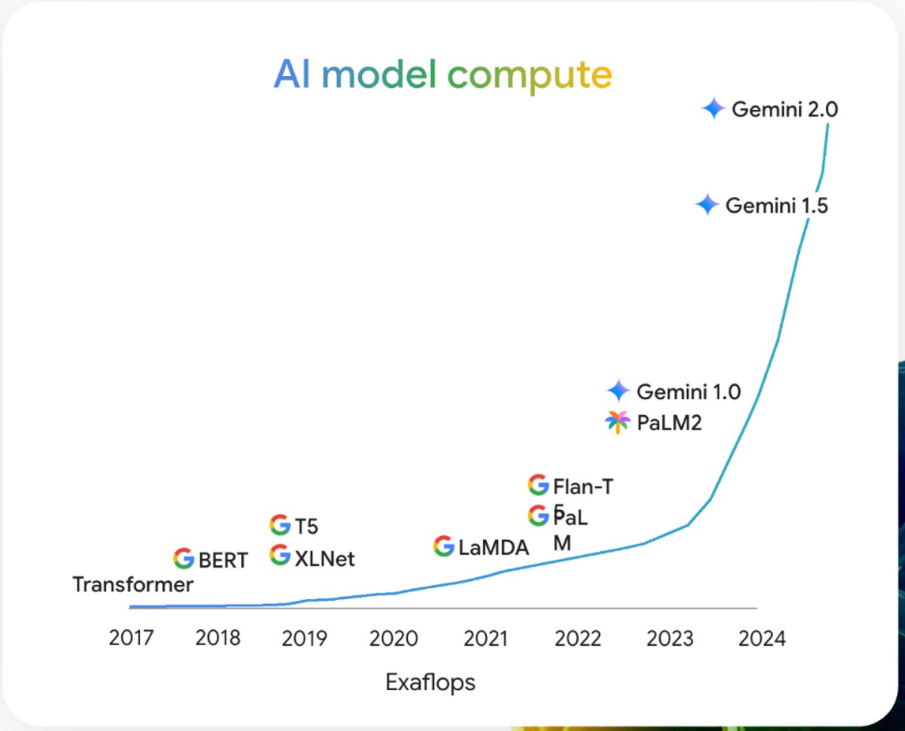
- Over the past few years, $/transistor has also flattened or even gone up with new process nodes



Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten
Dotted line extrapolations by C. Moore

As presented by Amin Vahdat, Google - 2025 USC Symposium

The demand for **ML compute** is growing exponentially



As presented by Amin Vahdat, Google - 2025 USC Symposium

Rapid innovation of performance-optimized hardware

TPU Supercomputing

TPU v1

2015
Internal inference accelerator

TPU v2

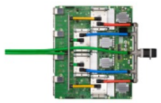
2018
Domain-specific AI supercomputing
256 chips
distributed shared memory

TPU v3

2020
Liquid cooled
1k chips distributed shared memory

TPU v4

2022
Optically reconfigurable
3D Torus
4k chips with distributed shared memory

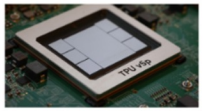


TPU v5e

2023
Purpose-built cost-efficiency and performance for medium/large-scale training and inference

TPU v5p

2023
Our powerful, scalable, and flexible AI accelerator



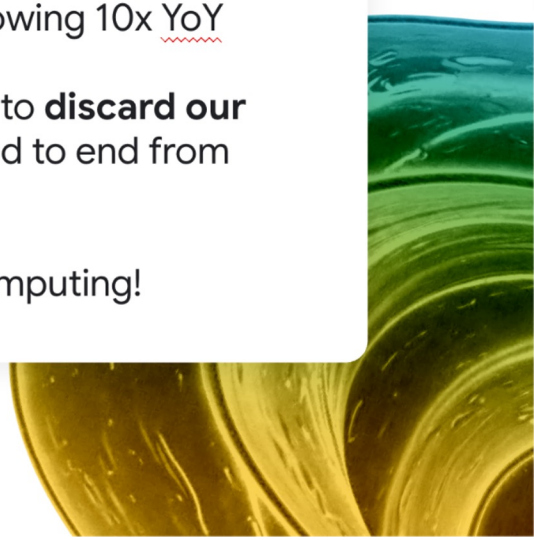
Trillium

2024
Designed for exceptional performance and efficiency. Enabling the next frontier of AI models

As presented by Amin Vahdat, Google - 2025 USC Symposium

Conclusions

- The last epoch of computing delivered sustained 2x performance for fixed cost, for decades, **transforming the computing landscape**
- This epoch of computing transitions us **from delivering data to delivering insights** but requires unprecedented levels of compute, with demand growing 10x YoY
- The techniques to meet this rising tide of demand will require us to **discard our developed conventional wisdom** and address the problems end to end from physics to algorithms and everything in between
- There has **never been a more exciting** time to be working in computing!



As presented by Amin Vahdat, Google - 2025 USC Symposium

ICONIQ

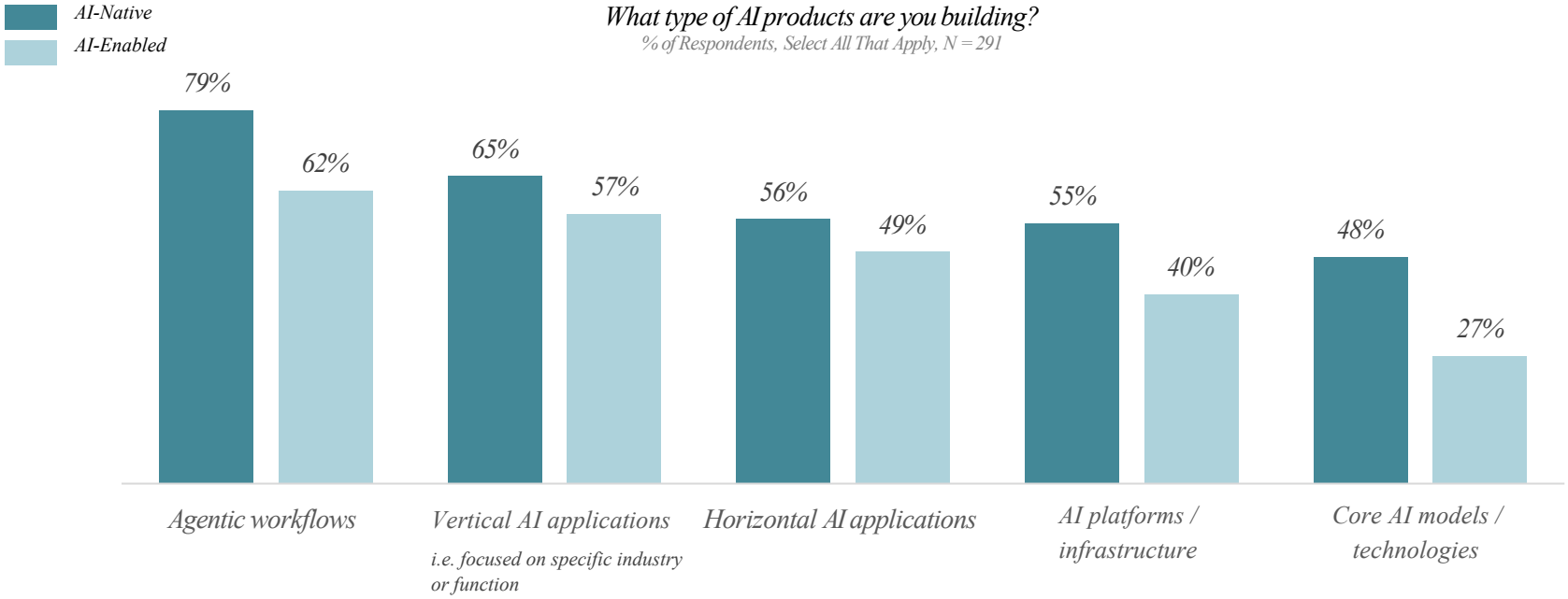
June 2025

The Builder's Playbook

2025 State of AI Report

Types of AI Products

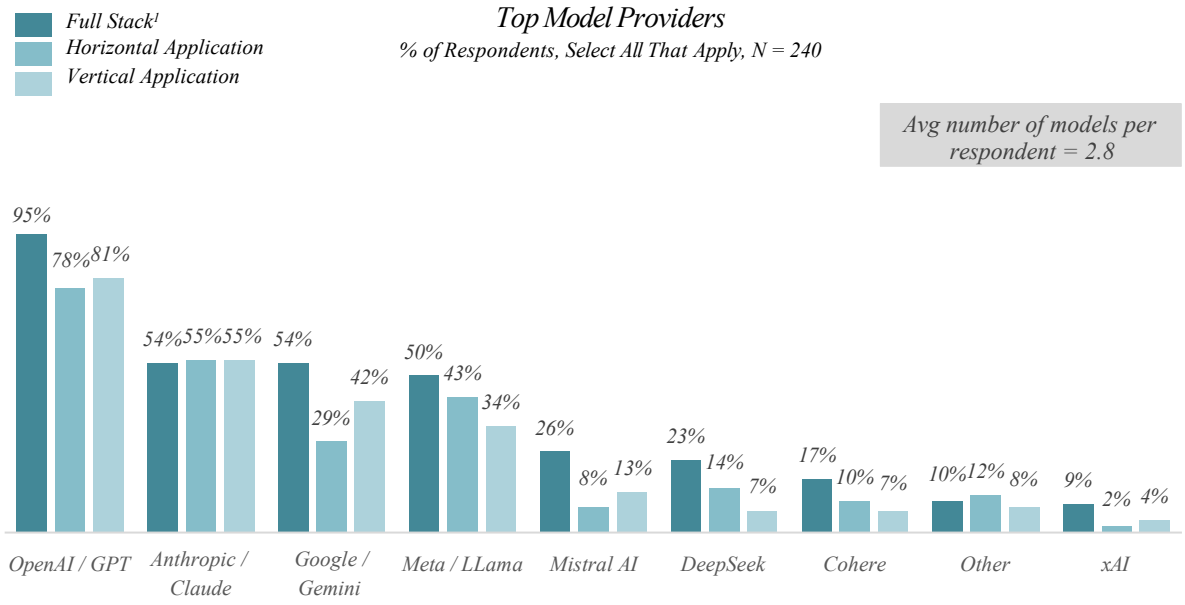
Agentic workflows and the application layer are the most common types of products being built across AI-native and AI- enabled companies; notably, around 80% of AI-native companies are currently building agentic workflows



Source: Perspectives from the ICONIQ GenAI Survey (April 2025) and perspectives from the ICONIQ team and network of AI leaders consisting of our community of CIO/CDOs overseeing AI initiatives in enterprises, CTOs, our Technical Advisory Board, and others in our network

Top Model Providers

OpenAI's GPT models continue to be the most popular model; however, many companies are increasingly adopting a multi-model approach to AI products across use cases



Notes: (1) Companies building both end user applications and core AI models/technologies

Source: Perspectives from the ICONIQ GenAI Survey (April 2025) and perspectives from the ICONIQ team and network of AI leaders consisting of our community of CIO/CDOs overseeing AI initiatives in enterprises, CTOs, our Technical Advisory Board, and others in our network

Companies are increasingly adopting a multi-model approach to AI products, leveraging different providers and models based on use case, performance, cost, and customer requirements.

This flexibility enables them to optimize for diverse applications like cybersecurity, sales automation, and customer service while ensuring compliance and superior user experience across regions.

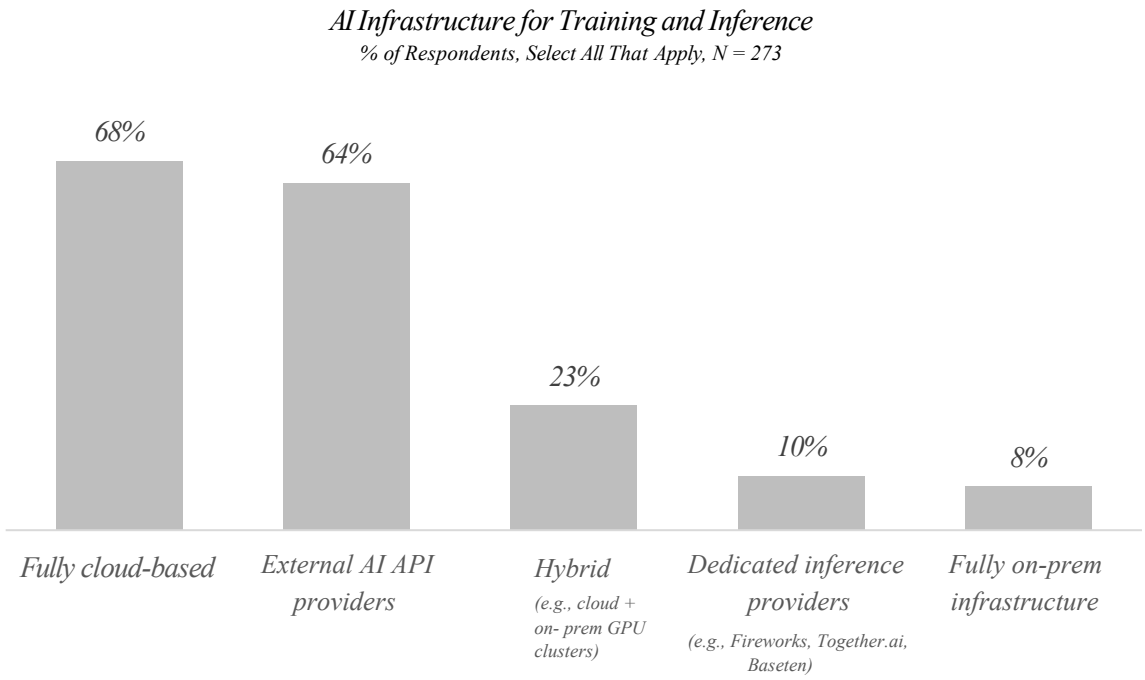
Architectures are being built to support quick model swaps, with some leaning toward open-source models for cost and inference speed advantages.

Generally, most respondents are using a combination of OpenAI models and 1-2 other models from the other providers.

> *We use different proprietary and 3rd party models because our customers have diverse needs. Specialized models allow us to better tailor the experiences for our customers and their use case -- sales automation, agents for customer service and internal tools. In addition, we can offer our customers more flexible price points and options, as well as be constantly experimenting with new models and business opportunities.*
> *VP Product, $1B+ Revenue, Full Stack AI Company*

AI Infrastructure

Most companies are using cloud-based solutions and AI API providers for training and inference



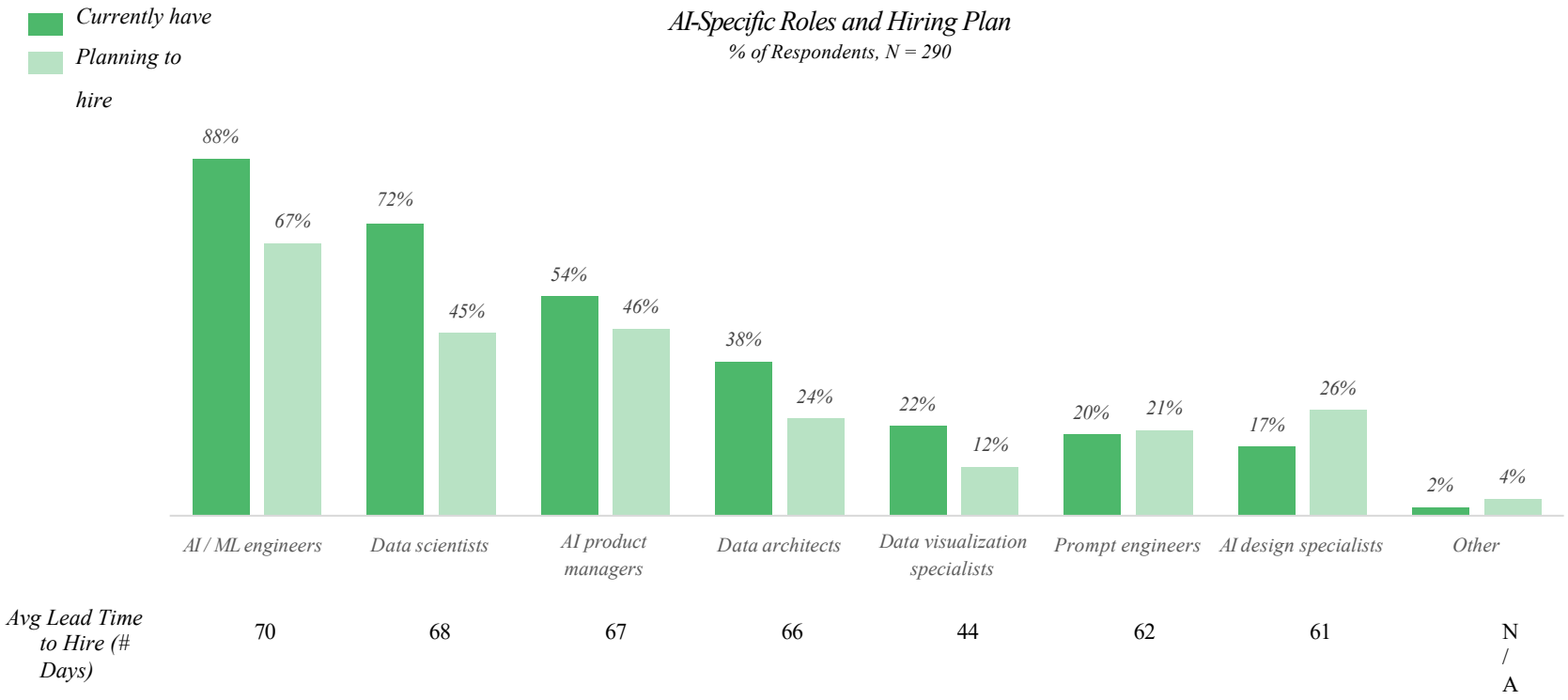
Most organizations are clearly leaning into fully managed AI solutions - 68% operate entirely in the cloud and 64% rely on external AI API providers - because this model minimizes upfront capital outlay and operational complexity, while maximizing speed-to- market. However, this reliance also means vendor selection, SLA negotiation, and cost-per-call management have become strategic priorities rather than just technical considerations.

Meanwhile, only 23% of teams use a hybrid approach and fewer than 1 in 10 maintain on-prem or dedicated inference infrastructure, underscoring that these models remain niche, adopted primarily in scenarios where control, compliance, or specialized performance justify the extra overhead. As real-time AI use cases grow, there's an emerging opportunity for turnkey inference platforms to capture more share, but any move away from fully managed services will hinge on a clear business case or regulatory imperative.

Source: Perspectives from the ICONIQ GenAI Survey (April 2025) and perspectives from the ICONIQ team and network of AI leaders consisting of our community of CIO/CDOs overseeing AI initiatives in enterprises, CTOs, our Technical Advisory Board, and others in our network

AI-Specific Roles

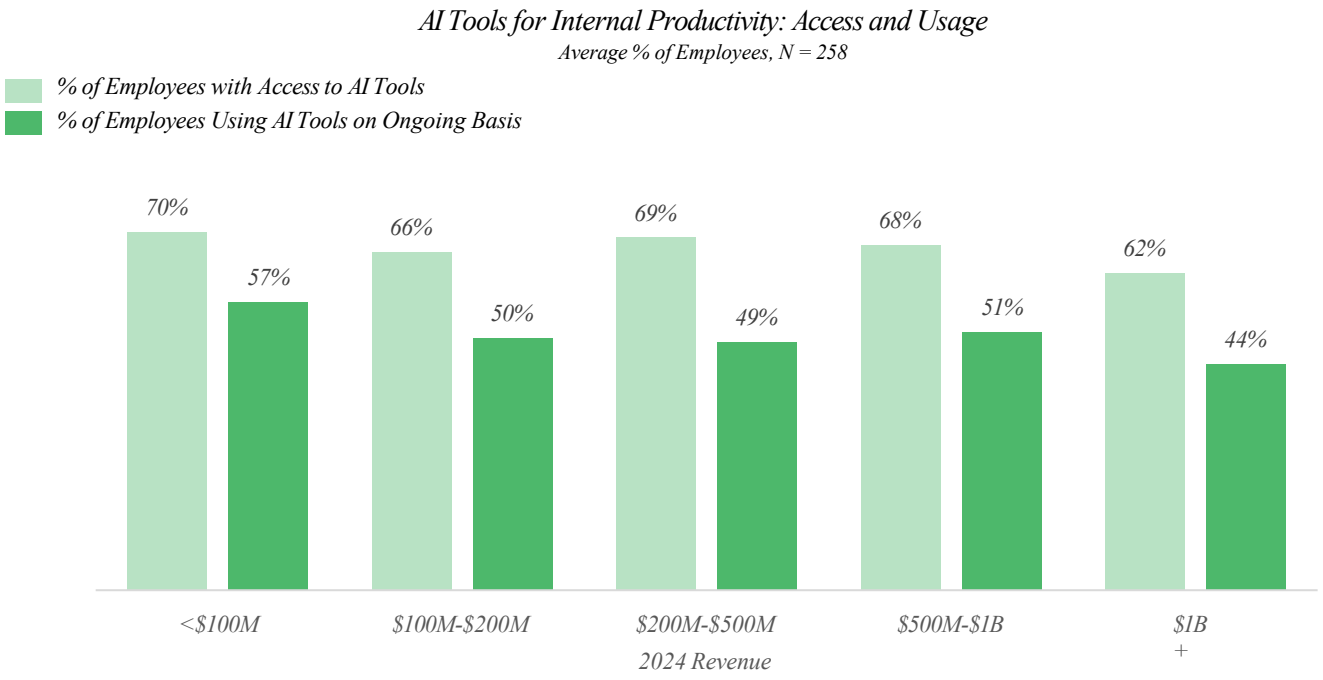
Most companies currently have dedicated AI/ML engineers, data scientists, and AI product managers, with AI/ML engineers taking the longest time on average to hire



Source: Perspectives from the ICONIQ GenAI Survey (April 2025) and perspectives from the ICONIQ team and network of AI leaders consisting of our community of CIO/CDOs overseeing AI initiatives in enterprises, CTOs, our Technical Advisory Board, and others in our network

AI Access and Usage

While around 70% of employees have access to various AI tools for internal productivity, only ~50% of employees are using AI tools on an ongoing basis with adoption more difficult in mature Enterprises ($1B+ revenue)



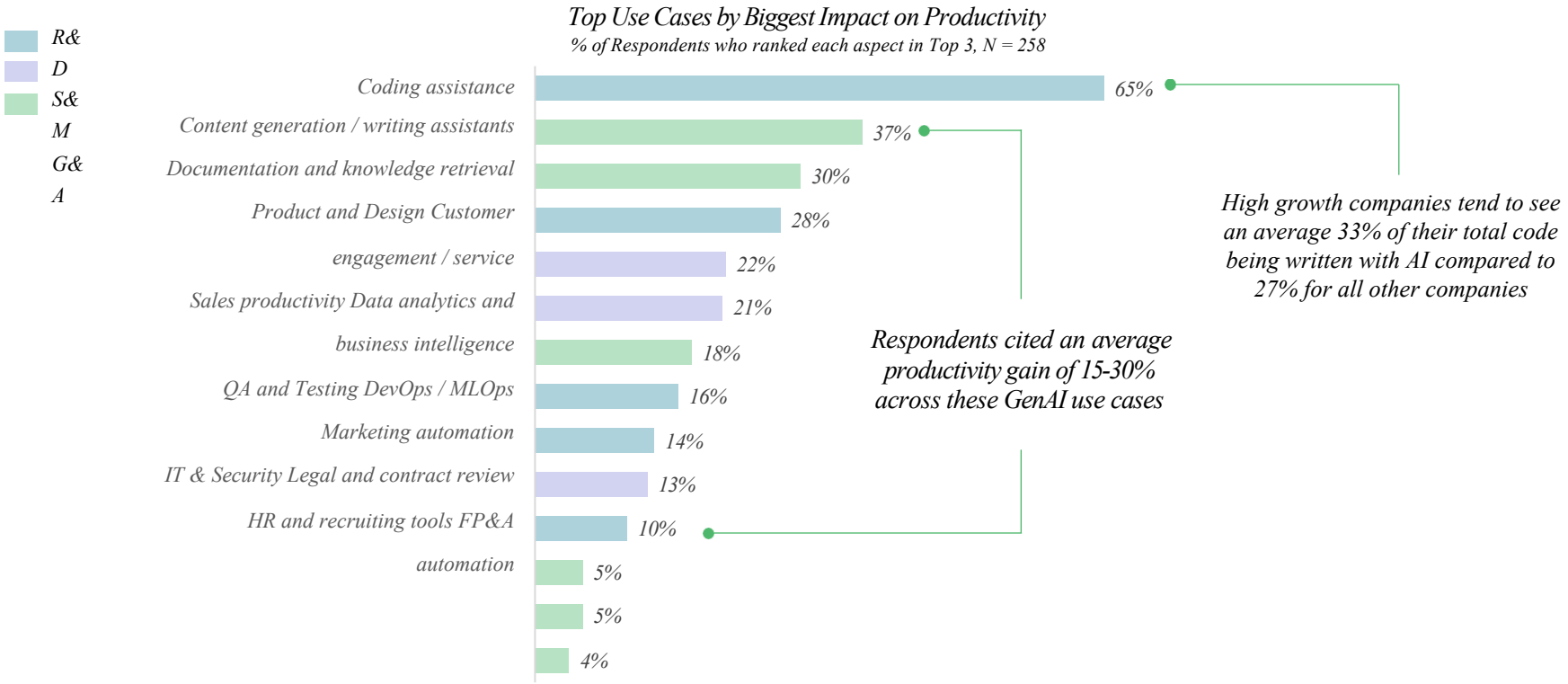
Don Vu
SVP, Chief Data & Analytics Officer, New York Life

Just deploying tools is a recipe for disappointment, particularly for large enterprises. To truly empower employees, you need to pair availability with scaffolding that includes training, spotlighting champions, and most importantly relentless executive support.

Source: Perspectives from the ICONIQ GenAI Survey (April 2025) and perspectives from the ICONIQ team and network of AI leaders consisting of our community of CIO/CDOs overseeing AI initiatives in enterprises, CTOs, our Technical Advisory Board, and others in our network

Top Use Cases: By Impact

Top use cases by impact mirror usage trends with coding assistance by far outpacing other use cases in terms of tangible impact on productivity



Source: Perspectives from the ICONIQ GenAI Survey (April 2025) and perspectives from the ICONIQ team and network of AI leaders consisting of our community of CIO/CDOs overseeing AI initiatives in enterprises, CTOs, our Technical Advisory Board, and others in our network

Most Used Tools: LLM & AI Application Development

Key Takeaways

Orchestration Frameworks Reign Supreme

•Top frameworks used include LangChain and Hugging Face's toolset which signals that teams clearly value high-level libraries that simplify prompt chaining, batching, and interfacing with either public or self-hosted models

•Around 70% of respondents also specified that they use private or custom LLM APIs

Safety and Higher-Level SDKs Gaining Traction

•Roughly 3 in 10 respondents use Guardrails to enforce safety checks, and almost a quarter leverage Vercel's AI SDK (23%) for rapid deployment which shows growing awareness that production LLM apps need both guardrails and streamlined integration layers

Long-Tail Experimentation

•Emerging players like CrewAI, Modal Labs, Instructor, DSPy, and DotTXT had weaker usage, indicating that while experimentation is widespread, broad standardization has yet to settle beyond the big players

Most Widely Used Tools

From survey respondents; By alphabetical order

Notes: Trademarks are the property of their respective owners. None of the companies illustrated have endorsed or recommended the services of ICONIQ.

Source: Perspectives from the ICONIQ GenAI Survey (April 2025) and perspectives from the ICONIQ team and network of AI leaders consisting of our community of CIO/CDOs overseeing AI initiatives in enterprises, CTOs, our Technical Advisory Board, and others in our network

Most Used Tools: Model Hosting

Key Takeaways

Direct-from-Provider Is King

- *The majority of teams hit model hosts directly via OpenAI, Anthropic, etc. underscoring that the path of least resistance remains calling the vendor's own inference APIs rather than building or integrating through a middle layer*

Hyperscalers Close Behind

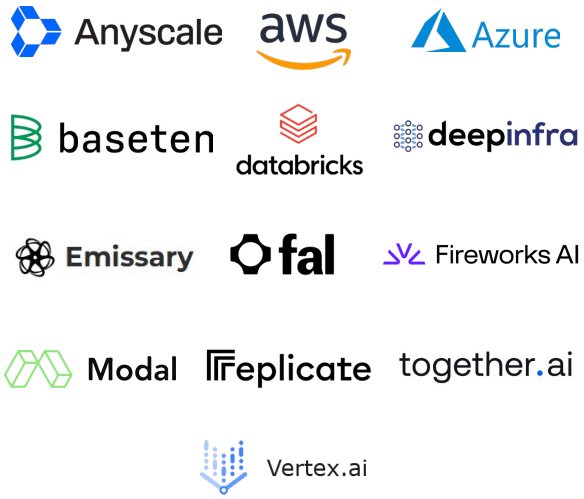
- *AWS Bedrock and Google Vertex AI have carved out substantial share, reflecting strong demand for unified, enterprise-grade ML platforms that bundle hosting with governance, security, and billing in a single pane*
- *In particular, a greater number of later-stage companies ($500M+ revenue) reported using hyperscaler solutions*

Fragmented Alternatives & Emerging Players

- *Beyond the big three, usage quickly fragments across players like Fireworks, Modal, Together.ai, AnyScale, Baseten, Replicate, Deep Infra, etc.*
- *This long tail suggests teams are still exploring specialty hosts, often driven by unique pricing, performance SLAs, or feature sets (e.g., custom runtimes, on-prem options)*

Most Widely Used Tools

From survey respondents; By alphabetical order



Notes: Trademarks are the property of their respective owners. None of the companies illustrated have endorsed or recommended the services of ICONIQ.
Source: Perspectives from the ICONIQ GenAI Survey (April 2025) and perspectives from the ICONIQ team and network of AI leaders consisting of our community of CIO/CDOs overseeing AI initiatives in enterprises, CTOs, our Technical Advisory Board, and others in our network

Most Used Tools: Coding Assistance

Key Takeaways

Dominance of First Movers

•GitHub Copilot is used by nearly three-quarters of development teams, thanks to its tight VS Code integration, multi-language support, and backing by GitHub's massive user base

•Copilot's network effects and product-market fit make it hard to dislodge, but the strong second-place showing for Cursor (used by 50% of respondents) signals appetite for diverse IDE integrations

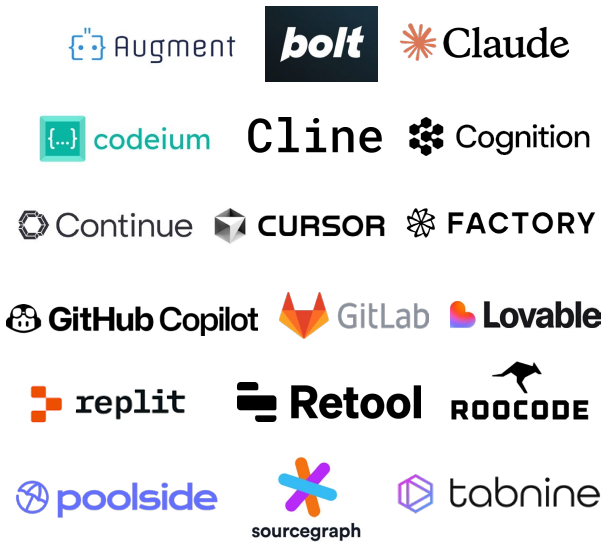
Long Tail of Offerings Lag

•After the top two, adoption drops off sharply with a fractured long tail of solutions, suggesting that while most teams have trialed at least one assistant, very few have standardized on alternatives

•Low-code or no-code solutions like Retool, Lovable, Bolt, and Replit also had honorable mentions indicating that there is increasing appetite in the market for idea-to-application solutions

Most Widely Used Tools

From survey respondents; By alphabetical order



Notes: Trademarks are the property of their respective owners. None of the companies illustrated have endorsed or recommended the services of ICONIQ.

Source: Perspectives from the ICONIQ GenAI Survey (April 2025) and perspectives from the ICONIQ team and network of AI leaders consisting of our community of CIO/CDOs overseeing AI initiatives in enterprises, CTOs, our Technical Advisory Board, and others in our network

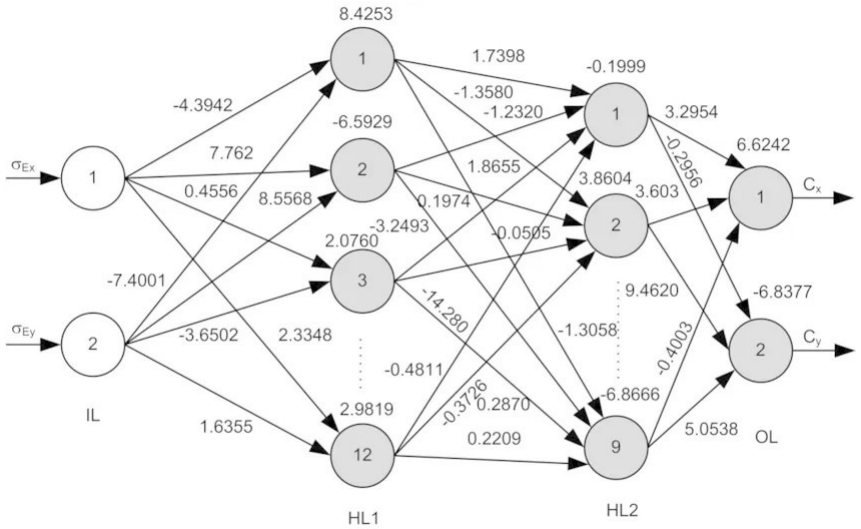
https://x.com/NathanLands/status/1936012661587951869

Software 1.0 = code

Software 2.0 = weights

Software 2.0

Andrej Karpathy Follow 9 min read · Nov 11, 2017



Software 1.0

computer code

programs

computer

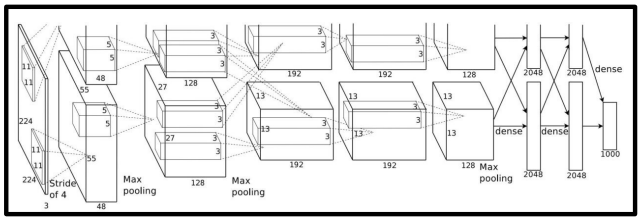
*became programmable in **~1940s***

Software 2.0

weights

programs

neural net



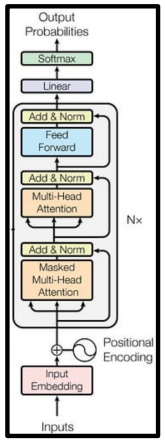
fixed function neural net
*e.g. AlexNet: for image recognition (**~2012**)*

Software 3.0

prompts

programs

LLM



~2019

LLM = programmable neural net!

Example: Sentiment Classification

Software 1.0

```python
def simple_sentiment(review: str) -> str:
    """Return 'positive' or 'negative' based on a tiny keyword lexicon."""
    positive = {
        "good", "great", "excellent", "amazing", "wonderful", "fantastic",
        "awesome", "loved", "love", "like", "enjoyed", "superb", "delightful"
    }
    negative = {
        "bad", "terrible", "awful", "poor", "boring", "hate", "hated",
        "dislike", "worst", "dull", "disappointing", "mediocre"
    }

    score = 0
    for word in review.lower().split():
        w = word.strip(".,!?;:")          # crude token clean-up
        if w in positive:
            score += 1
        elif w in negative:
            score -= 1

    return "positive" if score >= 0 else "negative"
```

Software 2.0

10,000 positive examples

10,000 negative examples

encoding (e.g. bag of words)

train binary classifier

parameters

Software 3.0

You are a sentiment classifier. For every review that appears between the tags
<REVIEW> ... </REVIEW>, respond with **exactly one word**, either POSITIVE or NEGATIVE (all-caps, no punctuation, no extra text).

Example 1
<REVIEW>I absolutely loved this film—the characters were engaging and the ending was perfect.</REVIEW>
POSITIVE

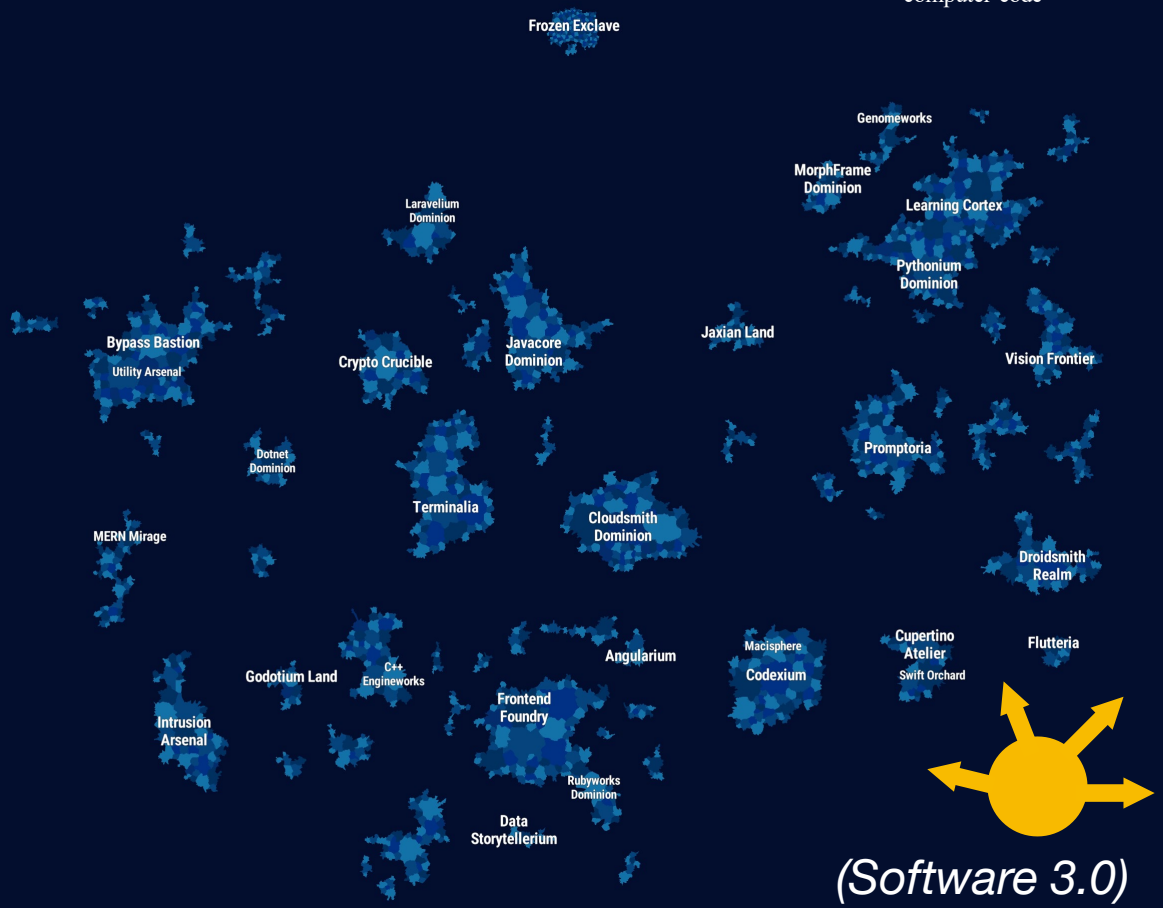
Example 2
<REVIEW>The plot was incoherent and the acting felt forced; I regret watching it.</REVIEW>
NEGATIVE

Example 3
<REVIEW>An energetic soundtrack and solid visuals almost save it, but the story drags and the jokes fall flat.</REVIEW>
NEGATIVE

Now classify the next review.

"Map of GitHub" (Software 1.0)

computer code

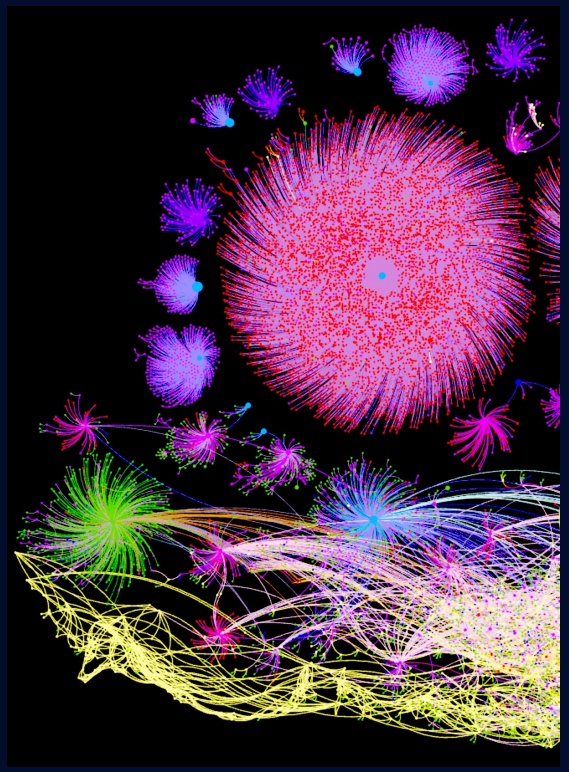


(Software 3.0)

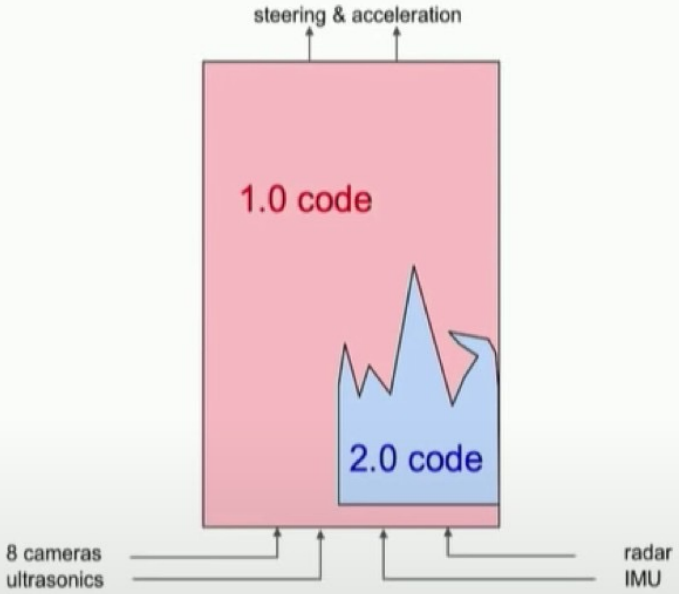
LLM prompts, in English

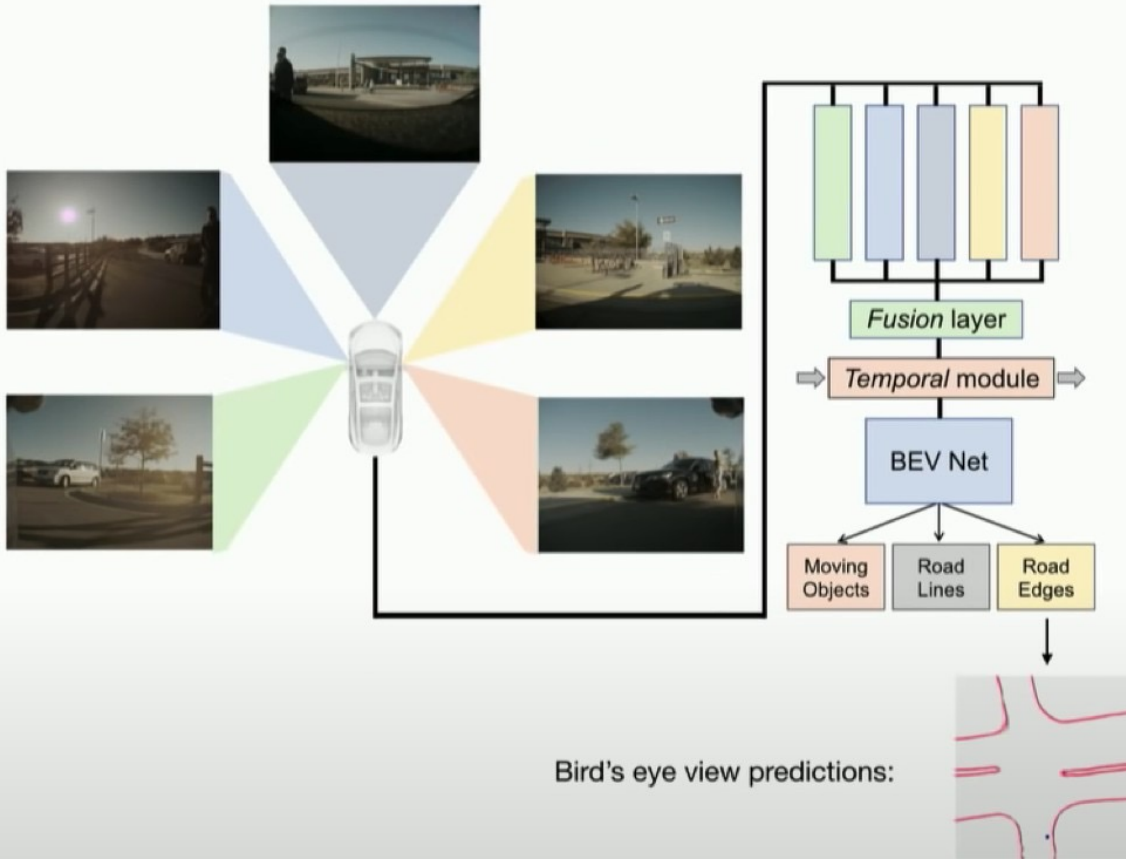
HuggingFace Model Atlas (Software 2.0)

neural network weights



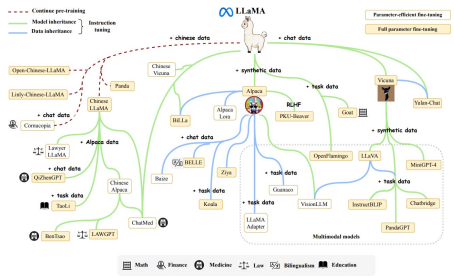
Software is eating the world
Software 2.0 eating Software 1.0



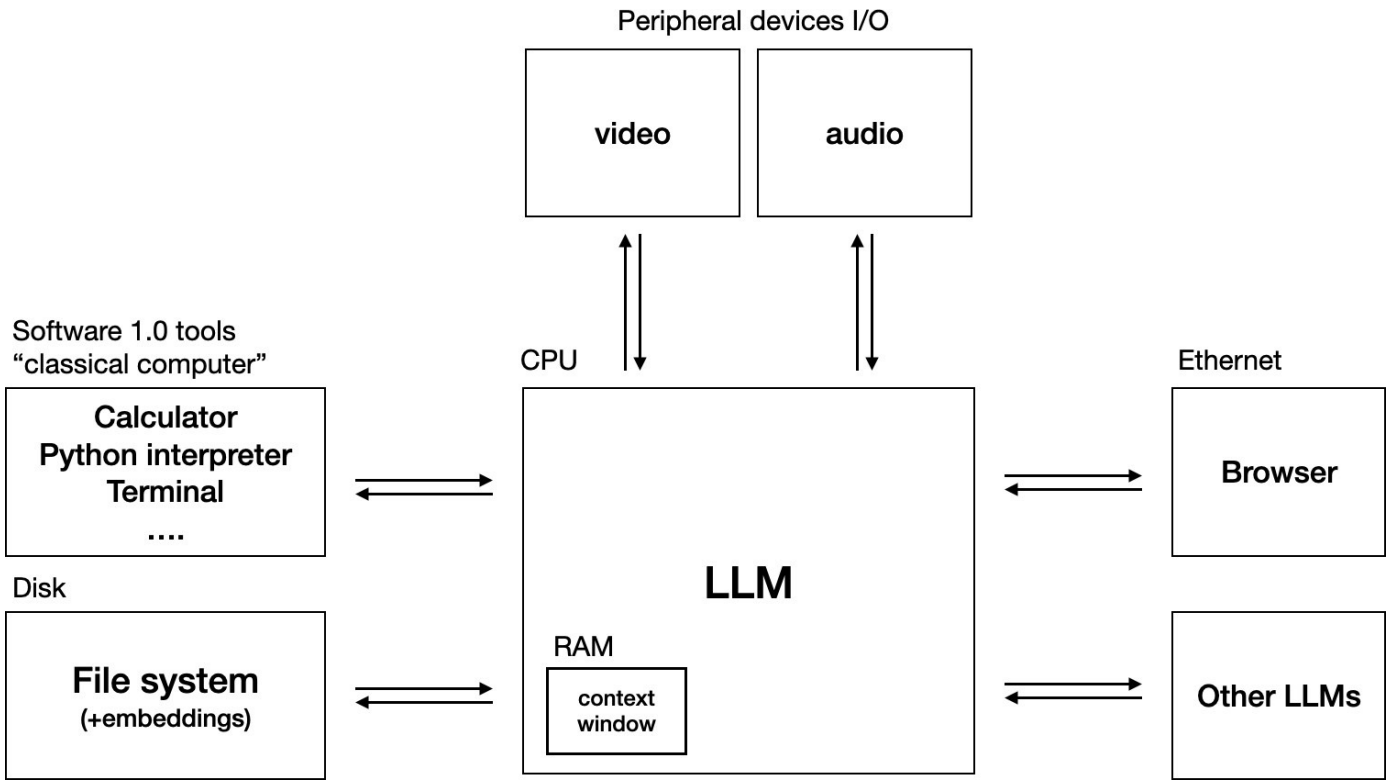


LLMs have properties of Operating Systems...

- *LLMs are increasingly complex software ecosystems, not simple commodities like electricity.*
- *LLMs are Software. Trivial to copy & paste, manipulate, change, distribute, open source, steal..., not physical infrastructure.*
- *Some amount of switching friction due to different features, performance, style, capabilities etc. per domain.*
- *System/user (prompt) space ~= kernel/user (memory) space*
- *...*

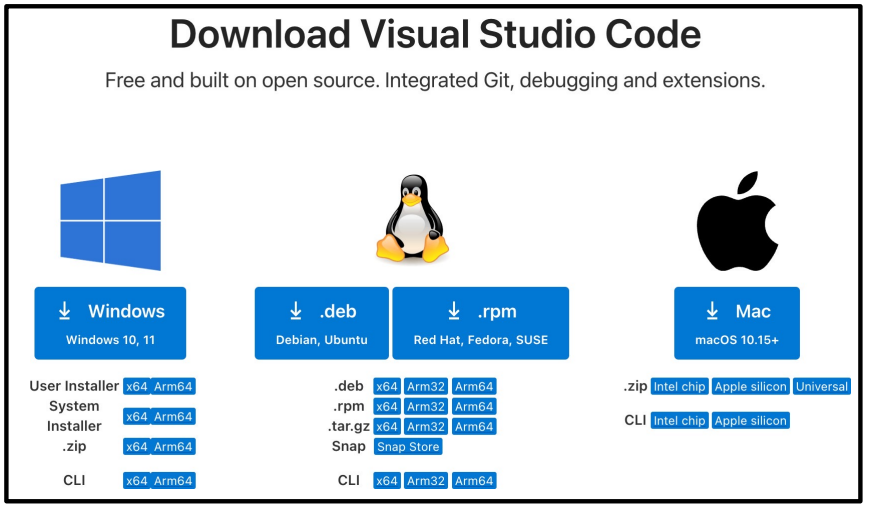


LLM OS


Peripheral devices I/O
video
audio
Software 1.0 tools
“classical computer”
CPU
Ethernet
Calculator
Python interpreter
Terminal
....
LLM
Browser
Disk
RAM
File system
(+embeddings)
context
window
Other LLMs

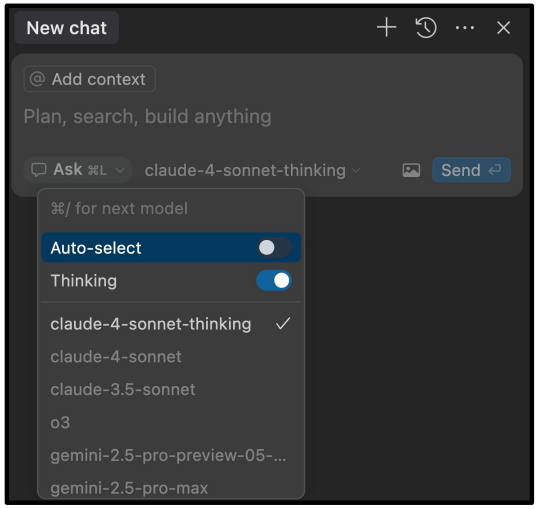
You can run an app like VS Code on:

- *Windows 10, 11*
- *Mac 10.15*
- *Linux*
- *...*



Just like you can run an LLM app like Cursor on:

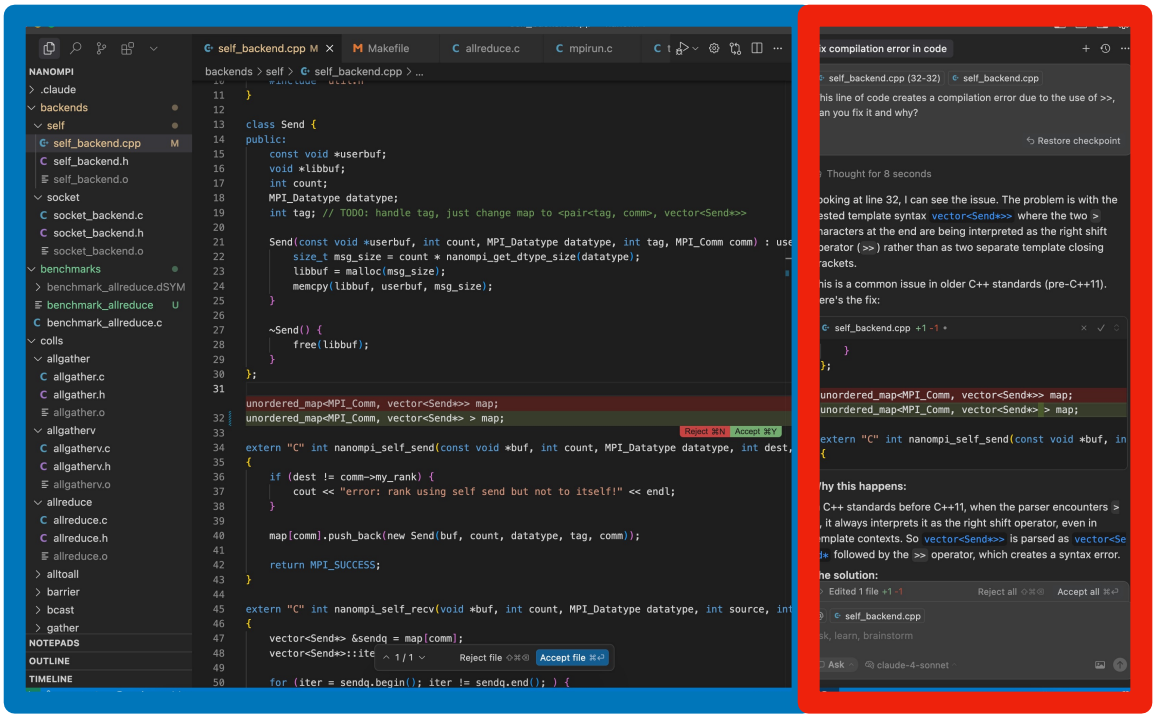
- *GPT o3*
- *Claude 4-sonet*
- *Gemini 2.5-pro*
- *DeepSeek*
- *...*



Example: Anatomy of Cursor

Traditional interface

LLM integration



1. Package state into a context window before calling LLM.

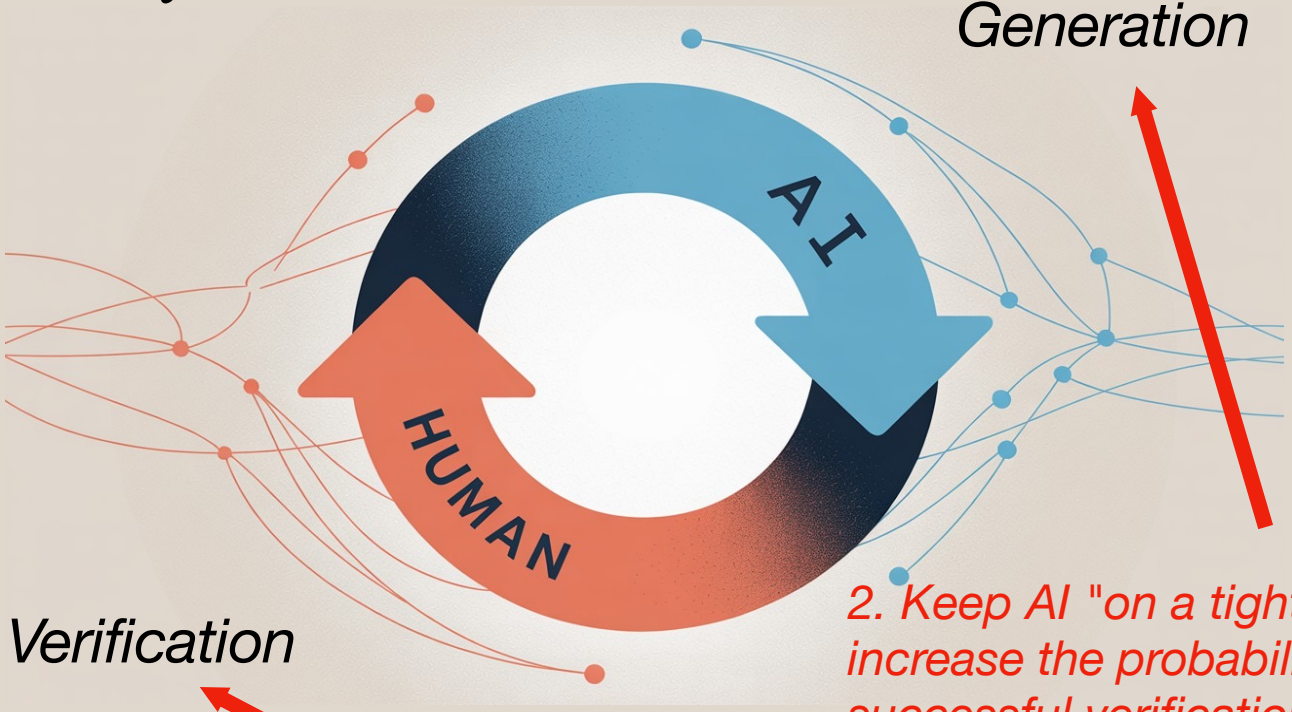
2. Orchestrate and call multiple models (e.g. embedding models, chat models, diff apply models, ...)

3. Application-specific GUI

4. Autonomy slider: Tab → Cmd+K → Cmd+L → Cmd+I (agent mode)

autonomy slider

Consider the full workflow of partial autonomy UIUX



Example: keeping agents on the leash

"AI-assisted coding" workflows (very rapidly evolving...)

- *describe the single, next concrete, incremental change*
- *don't ask for code, ask for approaches*
  - *pick an approach, draft code*
  - *review / learn: pull up API docs, ask for explanations, ...*
  - *wind back, try a different approach*
- *test*
- *git commit*
- *ask for suggestions on what could be implemented next*
- *repeat*

Example: keeping agents on the leash

Here's an example. This prompt is not unreasonable but not particularly thoughtful:

```
Write a Python rate limiter that limits users to 10 requests per minute.
```

I would expect this prompt to give okay results, but also miss some edge cases, good practices and quality standards. This is how you might see someone at nilenso prompt an AI for the same task:

```
Implement a token bucket rate limiter in Python with the following requirements:

- 10 requests per minute per user (identified by `user_id` string)
- Thread-safe for concurrent access
- Automatic cleanup of expired entries
- Return tuple of (allowed: bool, retry_after_seconds: int)

Consider:
- Should tokens refill gradually or all at once?
- What happens when the system clock changes?
- How to prevent memory leaks from inactive users?

Prefer simple, readable implementation over premature optimization. Use stdlib only (no Redis/external deps).
```

https://blog.nilenso.com/blog/2025/05/29/ai-assisted-coding/

Andrej Karpathy 
@karpathy

Promote  ···

There's a new kind of coding I call "vibe coding", where you fully give in to the vibes, embrace exponentials, and forget that the code even exists. It's possible because the LLMs (e.g. Cursor Composer w Sonnet) are getting too good. Also I just talk to Composer with SuperWhisper so I barely even touch the keyboard. I ask for the dumbest things like "decrease the padding on the sidebar by half" because I'm too lazy to find it. I "Accept All" always, I don't read the diffs anymore. When I get error messages I just copy paste them in with no comment, usually that fixes it. The code grows beyond my usual comprehension, I'd have to really read through it for a while. Sometimes the LLMs can't fix a bug so I just work around it or ask for random changes until it goes away. It's not too bad for throwaway weekend projects, but still quite amusing. I'm building a project or webapp, but it's not really coding - I just see stuff, say stuff, run stuff, and copy paste stuff, and it mostly works.

Vibe coding

From Wikipedia, the free encyclopedia

This article **may contain an excessive number of citations**. Please help remove low-quality or irrelevant citations. *(June 2025)* *(Learn how and when to remove this message)*

Vibe coding (or **vibecoding**) is an approach to producing software by using artificial intelligence (AI), where a person describes a problem in a few natural language sentences as a prompt to a large language model (LLM) tuned for coding. The LLM generates software based on the description, shifting the programmer's role from manual coding to guiding, testing, and refining the AI-generated source code.[1][2][3]

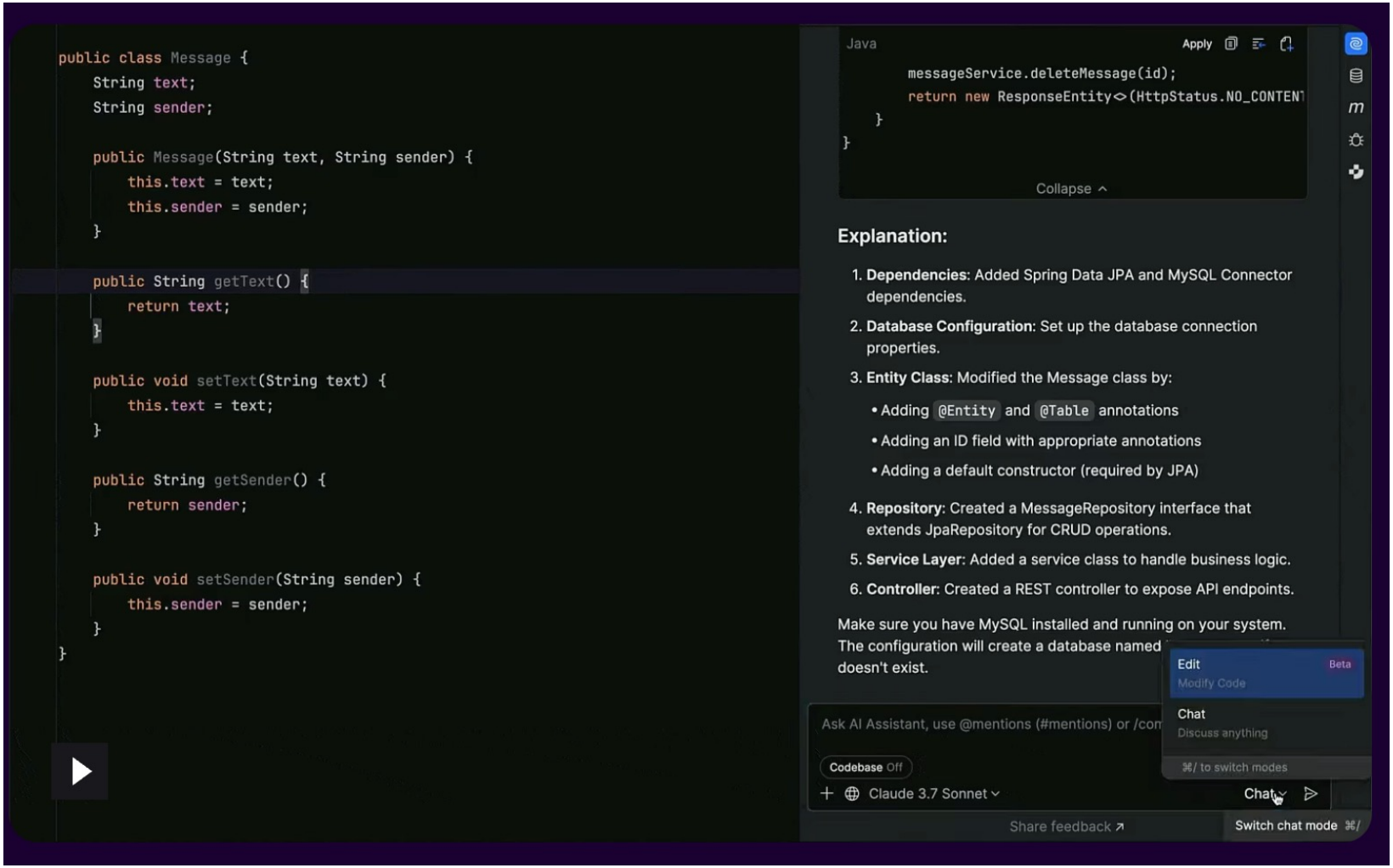
Advocates of vibe coding say that it allows even amateur programmers to produce software without the extensive training and skills required for software engineering.[4] The term was introduced by Andrej Karpathy in February 2025[5][2][4][1] and listed in the Merriam-Webster Dictionary the following month as a "slang & trending" noun.[6]

Example: Tesla Autopilot

autonomy slider

- *keep the lane*
- *keep distance from the car ahead*
- *take forks on highway*
- *stop for traffic lights and signs*
- *take turns at intersections*
- *...*

JetBrains AI Assistant



https://www.jetbrains.com/ai-assistant/

Top AI Stories of 2025 – Deeplearning.ai

- Thinking Models Solve Bigger Problems
  - Reasoning dramatically improves LLM performance, However, better output comes at a cost: millions more tokens.
- Big AI lures talent with huge pay (example: Meta's Zuckerberg)
- Data-Center buildout goes big ($300B of AI capital spending this year)
- Agents write code faster, cheaper
  - Boris Cherny: Claude Code update written with Claude Code
  - "While some observers worried that AI would replace junior developers, it turns out that developers who are skilled at using AI can prototype applications better and faster."
- China's AI Chip industry takes root (China's 50% subsidies for using Huawei chips

https://info.deeplearning.ai/top-stories-of-2025-big-ai-poaches-talent-reasoning-models-boost-performance-agents-write-code-data-centers-drive-gdp-china-turns-the-tables

2025 LLM Year in Review– Andrej Karpathy

- Ghosts vs. Animals / Jagged Intelligence
  - Human neural nets are optimized for survival of a tribe in the jungle, but LLM neural nets are optimized for imitating humanity's text, collecting rewards in math puzzles
  - LLMs display amusingly jagged performance characteristics - they are at the same time a genius polymath and a confused and cognitively challenged grade schooler
- Cursor / new layer of LLM apps
  - bundle and orchestrate LLM calls for specific verticals
  - provide an application-specific GUI for the human in the loop
  - offer an "autonomy slider"
- Claude Code / AI that lives on your computer
  - strings together tool use and reasoning for extended problem solving
  - runs on your computer and with your private environment, data and context
  - a little spirit/ghost that "lives" on your computer.

2025 LLM Year in Review– Andrej Karpathy (continued)

- Vibe coding
  - crossed threshold necessary to build impressive programs simply via English
  - “I coined the term "vibe coding" in this shower of thoughts tweet totally oblivious to how far it would go :).”
  - programming is not reserved for highly trained professionals; anyone can do it
  - it empowers trained professionals to write a lot more (vibe coded) software that would otherwise never be written.
- Nano banana / LLM GUI
  - in terms of the UIUX, "chatting" with LLMs is a bit like issuing commands to a computer console in the 1980s.
  - People actually dislike reading text; people love to consume information visually
  - who is actually going to build the LLM GUI? In this world view, nano banana is a first early hint of what that might look like

https://karpathy.bearblog.dev/year-in-review-2025/

AI Datacenters in Space – Starcloud and SpaceX

- **10x lower energy costs** — Nearly unlimited solar power in orbit eliminates reliance on terrestrial grid power; costs projected cheaper than land-based data centers even including launch expenses
- **No water needed for cooling** — Space's vacuum acts as an infinite heat sink; waste heat radiates into space via infrared, conserving Earth's freshwater resources
- **First data center-class GPU in orbit** — NVIDIA H100 launching November 2025 on the 60kg Starcloud-1 satellite, delivering 100x more compute than any previous space-based system
- **Real-time Earth observation** — In-space inference enables wildfire detection and distress response in minutes instead of hours; processes 10 GB/sec of synthetic-aperture radar data on-site
- **Massive scale planned** — 5-gigawatt orbital data center with solar/cooling panels ~4 km wide; CEO predicts "nearly all new data centers will be built in space" within 10 years
- **10x carbon savings** — Only environmental cost is launch; thereafter, 10x CO_2 reduction over the data center's lifetime compared to Earth-based equivalents

https://blogs.nvidia.com/blog/starcloud/

Nvidia's $20B Groq "Acquisition" (December 2025)

- **Largest Nvidia deal ever** — $20 billion in cash for Groq's IP and talent; nearly 3x Groq's $6.9B valuation from September; previous record was $7B Mellanox acquisition
- **Structured to avoid antitrust scrutiny** — Framed as "non-exclusive licensing agreement" rather than acquisition; analyst Stacy Rasgon: deal "keeps the fiction of competition alive"
- **Acqui-hire of key talent** — Founder/CEO Jonathan Ross (creator of Google's original TPU) and President Sunny Madra joining Nvidia; ~90% of Groq employees moving to Nvidia
- **Targets AI inference market** — Groq's LPU chips run LLMs at 10x speed, 1/10th energy of GPUs using on-chip SRAM; Nvidia dominant in training but faces competition in inference
- **Offense and defense play** — Integrates Groq's low-latency tech into Nvidia AI factory architecture while blocking AMD, Intel, and hyperscalers from acquiring a differentiated inference competitor

https://www.cnbc.com/2025/12/26/nvidia-groq-deal-is-structured-to-keep-fiction-of-competition-alive.html

Karpathy's Inspired Coding Guidelines

Author: **Jiayuan Zhang** @jiayuan_jy

A single .md file to improve AI Assistant behavior, derived from Andrej Karpathy's observations on LLM coding pitfalls.

The Problems

From Andrej's post:

> "The models make wrong assumptions on your behalf and just run along with them without checking. They don't manage their confusion, don't seek clarifications, don't surface inconsistencies, don't present tradeoffs, don't push back when they should."
>
> "They really like to overcomplicate code and APIs, bloat abstractions, don't clean up dead code… implement a bloated construction over 1000 lines when 100 would do."
>
> "They still sometimes change/remove comments and code they don't sufficiently understand as side effects, even if orthogonal to the task."

https://x.com/karpathy/status/2015883857489522876?s=20

Karpathy's Inspired Coding Guidelines (cont'd)

The Solution

Four principles in one file that directly address these issues:

Principle	Addresses
Think Before Coding	Wrong assumptions, hidden confusion, missing tradeoffs
Simplicity First	Overcomplication, bloated abstractions
Surgical Changes	Orthogonal edits, touching code you shouldn't
Goal-Driven Execution	Leverage through tests-first, verifiable success criteria

Four Principles in Detail

1. Think Before Coding

Don't assume. Don't hide confusion. Surface tradeoffs.

LLMs often pick an interpretation silently and run with it. This principle forces explicit reasoning:

- **State assumptions explicitly** — If uncertain, ask rather than guess
- **Present multiple interpretations** — Don't pick silently when ambiguity exists
- **Push back when warranted** — If a simpler approach exists, say so
- **Stop when confused** — Name what's unclear and ask for clarification

2. Simplicity First

Minimum code that solves the problem. Nothing speculative.

Combat the tendency toward overengineering:

- No features beyond what was asked
- No abstractions for single-use code
- No "flexibility" or "configurability" that wasn't requested
- No error handling for impossible scenarios
- If 200 lines could be 50, rewrite it

The test: Would a senior engineer say this is overcomplicated? If yes, simplify.

Four Principles in Detail (cont'd)

3. Surgical Changes

Touch only what you must. Clean up only your own mess.

When editing existing code:

- Don't "improve" adjacent code, comments, or formatting
- Don't refactor things that aren't broken
- Match existing style, even if you'd do it differently
- If you notice unrelated dead code, mention it — don't delete it

When your changes create orphans:

- Remove imports/variables/functions that YOUR changes made unused
- Don't remove pre-existing dead code unless asked

The test: Every changed line should trace directly to the user's request.

Four Principles in Detail (cont'd)

4. Goal-Driven Execution

Define success criteria. Loop until verified.

Transform imperative tasks into verifiable goals:

Instead of…	Transform to…
"Add validation"	"Write tests for invalid inputs, then make them pass"
"Fix the bug"	"Write a test that reproduces it, then make it pass"
"Refactor X"	"Ensure tests pass before and after"

For multi-step tasks, state a brief plan:

```
1. [Step] → verify: [check]
2. [Step] → verify: [check]
3. [Step] → verify: [check]
```

Strong success criteria let the LLM loop independently. Weak criteria ("make it work") require constant clarification.

Four Principles in Detail (cont'd)

Key Insight

From Andrej:

> "LLMs are exceptionally good at looping until they meet specific goals…
> Don't tell it what to do, give it success criteria and watch it go."

The "Goal-Driven Execution" principle captures this: transform imperative instructions into declarative goals with verification loops.

How to Know It's Working

These guidelines are working if you see:

- **Fewer unnecessary changes in diffs** — Only requested changes appear
- **Fewer rewrites due to overcomplication** — Code is simple the first time
- **Clarifying questions come before implementation** — Not after mistakes
- **Clean, minimal PRs** — No drive-by refactoring or "improvements"

Four Principles in Detail (cont'd)

Customization

These guidelines are designed to be merged with project-specific instructions. Add them to your existing .md or create a new one.

For project-specific rules, add sections like:

```
## Project-Specific Guidelines

- Use TypeScript strict mode
- All API endpoints must have tests
- Follow the existing error handling patterns in src//errors.ts`
```

Tradeoff Note

These guidelines bias toward **caution over speed**. For trivial tasks (simple typo fixes, obvious one-liners), use judgment — not every change needs the full rigor.

The goal is reducing costly mistakes on non-trivial work, not slowing down simple tasks.

AI-Assisted Coding Guidelines .md File