AI Agents for Knowledge Workers

What they are, how they work, and how to work well with them

SEAN DAVIS, MD, PHD · 2026-07-22

You already know the hard part

You use chatbots. You write prompts. You judge whether the answer is any good.

That intuition transfers directly. This session adds three things:

- what changes when the AI can **act in your files**, not just chat
- what is actually happening under the hood — **context** and **tokens**
- a few habits that make these tools **safer, cheaper, and sharper**

Speaker notes

Anchor to what they already do. Nearly everyone in the room uses ChatGPT/Claude/Gemini as a chat window. The goal today is not “learn to program” — it’s to understand a new interface and a handful of mental models. Stay tool-agnostic throughout: the concepts hold across every major tool, and the handout covers specific options.

PRESENTER KEYS — s speaker view · f fullscreen · o overview · g go-to-slide c draw on this slide · b blank board · x cycle ink colour · del clear slide backspace clear all · d download drawings · q laser pointer · ? all keys (Drawings live only in this browser session unless you press d.)

TIMING (50-60 min slot). The deck runs long if you show everything; that is deliberate, so you can pick per audience. Planned shape: Parts 1-3 + demo ……. ~35 min (demo ~10 of it) Part 4 (extensions) …… ~8 min Where this is going …… ~7 min Wrap-up + Q&A ……. ~10 min

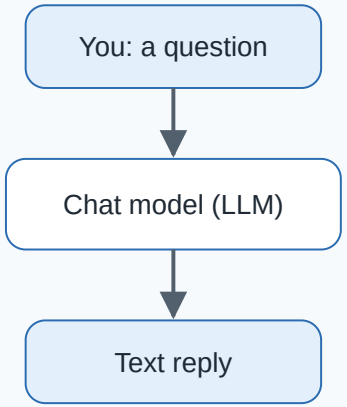
If you are behind after the demo, cut in this order: 1. “Model vs. agent” (Part 1) — nice-to-have, not load-bearing 2. “Shell and tools” (Part 4) — this audience already knows it; the overview figure covers it 3. “Multi-agent systems” (co-scientist / Virtual Lab) — keep Biomni and the two cancer papers, which are closer to home 4. “What a good session looks like” — it restates the takeaways slide Never cut: the demo punchline (check a row), privacy/PHI, or takeaways.

Part 1 — From chatbot to agent

Two ways to use the same model

Chatbot

answers from a prompt window

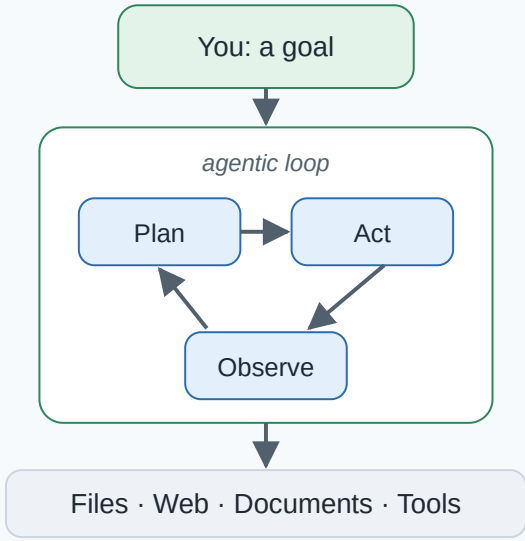


✗ no access to your files

copy · paste · adjust · repeat

AI agent

acts in your environment



✓ reads & changes your real files

Speaker notes

The model is the same kind of thing in both columns. The difference is the scaffolding around it. A chatbot answers from a prompt window — you copy, paste, adjust, repeat. A coding agent runs a loop: plan an action, call a tool (read a file, run a command), observe the result, and continue — and it does this inside your actual repository, with permissions.

Your work is file-shaped

So much of research life lives in files:

- manuscripts, grant applications, protocols, reviews
- folders of PDFs — papers, applicant CVs, reports
- sample sheets, metadata tables, marker-gene lists, data dictionaries
- analysis scripts and notebooks; slide decks and meeting notes

A chatbot reasons about a *snippet you paste*. An agent reasons about the *documents as they actually are*.

> “Take this list of 40 DOIs, fetch the open-access PDFs, extract the platform and cell counts from each, and flag the two whose methods disagree with their abstracts.”

Speaker notes

This is the key distinction worth repeating. The unit of work changes from “answer a question” to “operate on the real material.” Nobody in this room lacks a folder of PDFs they wish were a table. That’s the hook.

What agents do for knowledge work

Deep research

Read many sources and synthesize a cited summary.

e.g. compare 10 methods in a table

PDF extraction

Pull structured data out of stacks of PDFs.

e.g. 40 applicant CVs → a ranked CSV

Slides & decks

Draft and edit presentations from your notes.

e.g. these very slides

Document editing

Revise grants, manuscripts, and theses.

e.g. tighten this Aims page; flag gaps

Brainstorming

Think with your papers and decks as context.

e.g. from 3 papers, propose experiments

Same idea as code: give it the **real material** — PDFs, data, drafts, slides — not a description of it.

Speaker notes

This is the center of the talk. Walk each card slowly — each one is a real prompt pattern:

- Deep research: “Search PubMed for recent work on CAF subtypes in pancreatic cancer; read the top sources and write a one-page synthesis with citations, noting where they disagree.”
- PDF extraction: “These 40 postdoc applicant CVs are PDFs — extract degree, methods experience, and first-author papers, score against rubric.md, and give me a ranked CSV.” (Same loop as the paper demo, different folder.)
- Slides: “Turn outline.md into a 12-slide deck with speaker notes” — exactly how this deck was built.
- Document editing: “Tighten this Aims page, flag claims needing citations, and list inconsistencies with the project summary.”
- Brainstorming: “Given these three papers and my draft, propose follow-up experiments and push back on my weakest framing.”

The thread: give it the real material, not a description of the material.

The right mental model

A **capable, fast, tireless trainee** — not an oracle.

- reads quickly, drafts well, follows instructions, never gets bored
- also: makes mistakes, needs clear direction, benefits from supervision

You are the attending. **You sign the note.**

- from doing every task → **delegating with clear instructions**
- from remembering context → **providing context explicitly**
- from working alone → **reviewing output before it counts**

Speaker notes

The trainee/attending framing lands for this audience better than any software metaphor. It calibrates frustration (a wrong assumption usually means unclear instructions, not a broken tool) and it carries the responsibility message without moralizing: supervision is not optional, and the signature is yours. This framing returns on the privacy/trust slide.

Model vs. agent — one quick distinction

- The **model** (the LLM) is the reasoning engine: reads, plans, writes.
- The **agent** is the layer around it: tools, memory, permissions, a loop.

Different products are mostly different *agent layers* over similar models. Learn the concepts once; the skills transfer across tools.

Speaker notes

Deliberately tool-agnostic. Name a few examples verbally if asked (Claude, ChatGPT, Gemini all now ship agent modes; terminal tools exist for developers) but don’t put logos on the slide — the point is that “which model” and “which product” are separate questions, and this talk teaches the durable layer.

This separation is also what makes the local-model option on the privacy slide possible: because the agent and the model are different layers, you can keep the workflow and swap in a model running on your own hardware.

Live demo — let’s watch one work

The task I actually had

I needed ten papers on one topic — **real, open access, and downloadable** — to demo on.

By hand that is an afternoon: search, skim, check each licence, download, give up around paper six, and end with a folder nobody else can reuse.

So the preparation for this demo *was* the demo:

1. **Ask** a literature database for candidates
2. **Keep** only the ones that actually download
3. **Curate** for a real disagreement
4. **Save the list**, not the PDFs

Speaker notes

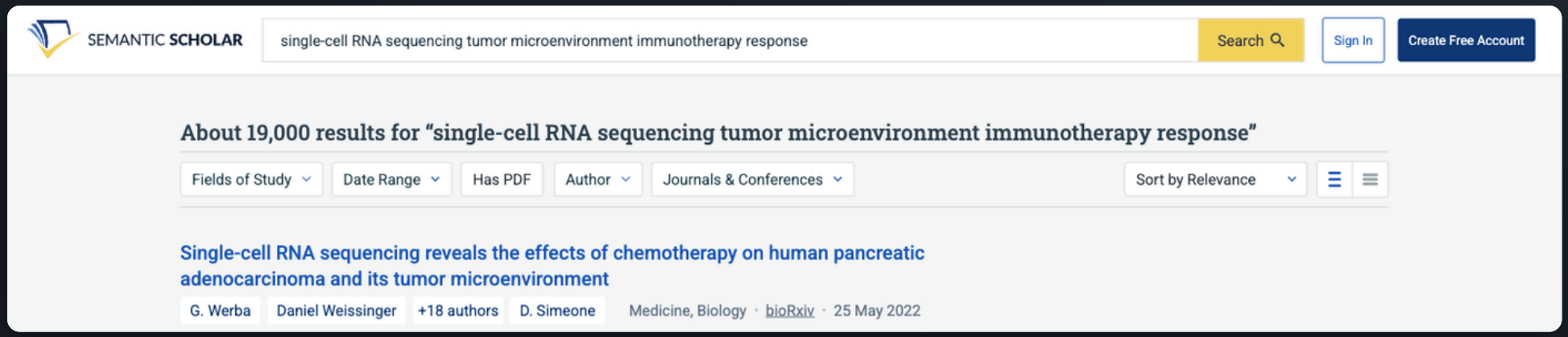
This slide exists because the demo otherwise looks like it started from a tidy CSV that fell from the sky. It didn’t — and the messy part is the part worth teaching.

Step 2 is the one to lean on. “Open access” in a database record does not mean “I can fetch it”: four publishers in my candidate set (AACR, BMJ, Wiley, Science) return 403 to a scripted request. The agent found that by trying every link and reporting the failures — exactly the verification habit from the mental-model slide.

Step 3 was mine, not the agent’s: I chose papers that disagree with each other so the synthesis step has something real to find. Say that out loud — the agent fetched and checked; deciding what makes a good teaching set was a human judgment.

Step 4 is why the repo holds a 3 KB list instead of 114 MB of PDFs.

Where the candidates came from



semanticscholar.org — free, ~200M papers. Beyond title and abstract, every record carries the DOI and says whether an **open-access PDF exists, where it is, and under which licence**.

This page is laid out for **human eyes**. Your agent would rather have the same search as **data**.

Speaker notes

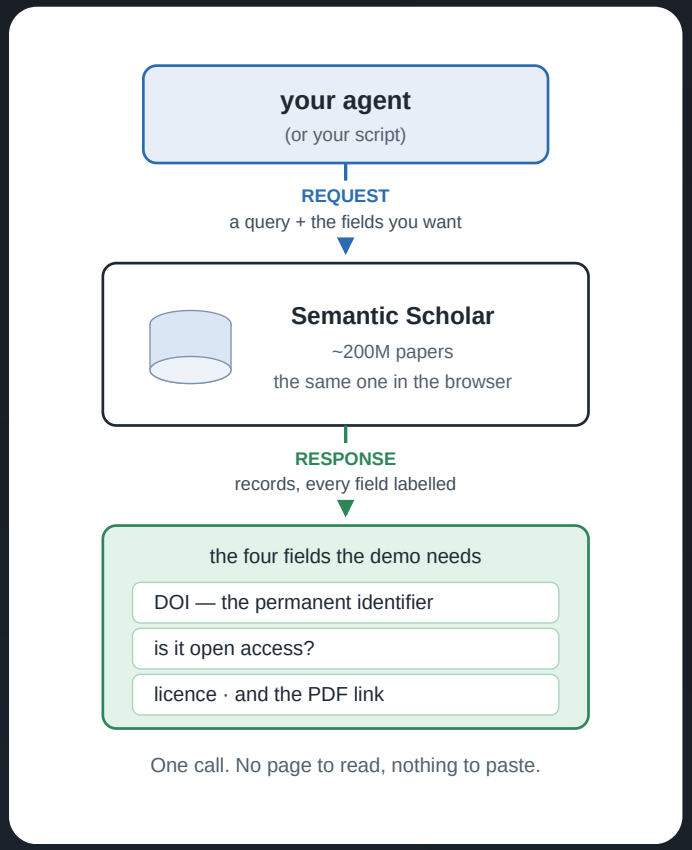
This is the human view — the same site anyone in the room could use by hand, and worth knowing about on its own if they don’t.

The top hit on that screen is literally the first paper in our CSV. Point at it.

The field that matters for what comes next is the open-access one. Most literature search tools tell you a paper exists; this one tells you whether you can actually get the file, and under what licence. That is the difference between a reading list and something an agent can act on.

Every search is also an API call

A website is built for eyes. An **API** is the same data served for programs — you send a query, you get back records with labelled fields.




```
{ "title": "Tumor microenvironment remodeling
           after neoadjuvant immunotherapy…",
  "year":  2023,
  "venue": "Genome Medicine",
  "externalIds": {
      "DOI": "10.1186/s13073-023-01164-9" },
  "isOpenAccess": true,
  "openAccessPdf": {
      "url":     "https://genomemedicine…",
      "license": "CCBY" } }
```

No scraping, no copy-paste — and **that record is row 1 of our CSV.**

Speaker notes

This is the slide for the half of the room who have never thought about an API. Keep the definition concrete: a website is for eyes, an API is the same thing for programs. Every field is labelled, so a program can use it without guessing.

Walk the four fields on the left, then point at them in the real response on the right: the DOI is the permanent identifier; isOpenAccess and license say whether we may take it; openAccessPdf.url is the actual file. Those four are the entire reason the demo works.

Say plainly that this response is real and lightly trimmed — the ellipses are mine, for the slide.

Practical, if asked: Semantic Scholar’s API is free and needs no key for light use; it rate-limits instead, which I hit while preparing this, and keys are free for heavier work. PubMed and Europe PMC have equivalent APIs, so the pattern generalises to whatever corpus you care about.

And the connection forward: this is the MCP idea from Part 4 arriving early. I never taught the agent what Semantic Scholar is — it had a tool for it, and the tool speaks this.

If someone asks “couldn’t I just ask a chatbot for ten papers?” — yes, and some of them would not exist. Going through a literature API is what makes every record a real one with a resolvable DOI.

What we start with: a reference list

`papers.csv` — 10 open-access single-cell papers on the tumor microenvironment and immunotherapy response.

doi	year	journal	short_name
10.1186/s13073-023-01164-9	2023	Genome Medicine	nsclc-neoadjuvant-ici
10.1038/s41590-022-01215-0	2022	Nature Immunology	gbm-immune-evolution
10.7150/thno.60540	2022	Theranostics	gastric-proinvasive-caf
…			*(7 more)*

No PDFs yet. Just DOIs — the way a reference list actually arrives.

Speaker notes

Start here deliberately: this is what people really have — a citation list, a saved search, a collaborator’s email. The PDFs are 114 MB; the list is 3 KB. Say that out loud, it lands.

The papers are all CC-BY / CC-BY-NC open access, deliberately: nothing paywalled, nothing sensitive. Same reason the privacy slide comes later.

Act 1 — “go get them”

> “Read `papers.csv`. Download each open-access PDF into `papers/`, named by `short_name`. Then tell me which ones failed and why.”

Watch the loop: read the list → fetch → **check it’s actually a PDF** → report what broke.

Speaker notes

This act is short (~2 min) but it earns the whole talk. Points to narrate:

- It handles a *list*, one row at a time — the same loop as any repetitive task.
- Publishers differ: four in the candidate set (AACR, BMJ, Wiley, Science) return 403 to a scripted request, so those papers never made the list — see the previous slide. Real friction, real workaround; that is what “verify” means in practice.
- It reports failures instead of silently producing a short table. Contrast with a chatbot, which would just hand you plausible-looking text.

If wifi is slow, the PDFs are already in papers/ from rehearsal — the agent will say “already present” and move on, which is itself a nice touch.

Act 2 — “now read them”

> “Read the PDFs in `papers/`. Build `extraction.csv` — cancer type, platform, number of cells, key finding, main limitation. Then write a one-page synthesis: where do these papers agree, and where do they disagree? Cite by DOI.”

Same loop, bigger job: plan → open a paper → extract → next → synthesize.

Speaker notes

The main act (~6-7 min). Narrate that it is reading files the audience can see in the folder, not recalling training data.

Watch for on-screen: it opens each PDF in turn; the row count grows; the review paper (scrnaseq-cancer-review) has no sample size, so a good agent flags it rather than inventing one. That’s the tell worth pointing at.

The real disagreement in this set — what predicts response to checkpoint blockade: a stemness signature, a T-cell exhaustion signature, CAF subpopulations, or myeloid composition? Four papers, four answers, all defensible. If the synthesis surfaces that tension, say so. If it papers over it, say that too — a smooth summary that hides a real disagreement is exactly the failure mode to watch for.

While it works, take audience questions.

The punchline: check a row

Open `extraction.csv`. Pick one row. Open that PDF. **Verify.**

- number of cells → is that the number in the paper, or the number in the abstract?
- “main limitation” → the authors’, or the agent’s opinion?

You would do this to a trainee’s first draft. Do it here.

Speaker notes

Do this live, on one row, in front of them — this is the single most important 90 seconds of the talk. Whether the row is right or wrong, the habit is the lesson. If it’s wrong, even better: that’s the talk’s thesis made concrete.

Fallbacks, in order: (1) phone hotspot if venue wifi fails; (2) the pre-recorded screen capture of the same run; (3) screenshots of the finished CSV + synthesis. Keep rehearsal outputs in /demo/backup/ either way.

What you just saw

- It went from a **reference list to a synthesis** without you touching a PDF
- It **planned**, then worked one paper at a time — not one giant answer
- Every step was **visible and interruptible**
- The output landed **in files you keep** — a CSV and a document, not a chat bubble
- And we checked a row before believing it — **that's the job now**

Speaker notes

Bridge slide: convert the demo into the concepts. The visible loop is Part 1; what it cost and why it sometimes misses things is Part 2; the habits that make it reliable are Part 3. If the demo failed and you used the recording, this slide still works verbatim.

Part 2 — What’s under the hood

The model reads tokens, not words

A model reads **tokens** — short chunks of text, not whole words:

Normal | ize | the | T | SV | columns | , | then | write | out | . = 11 tokens

1 token ≈ 4 characters ≈ ¾ of a word → 100 tokens ≈ 75 words

Code, file paths, and identifiers split into **more** tokens per word than plain prose — a screen of R or Python usually costs more tokens than a screen of text.

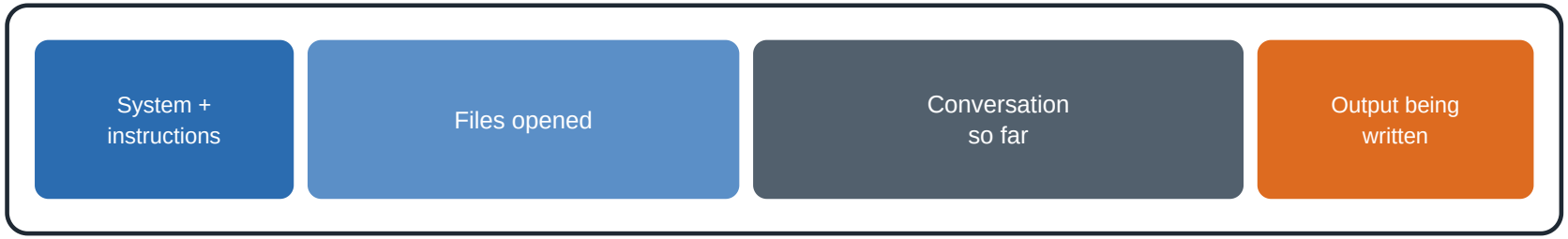
Speaker notes

Tokens are the unit the model actually processes — roughly ¾ of a word. Two practical takeaways for this audience: (1) documents are bigger than they look — a 30-page PDF is tens of thousands of tokens; (2) pricing and limits are all denominated in tokens, and generating output costs several times more than reading input. That’s the whole cost model in one sentence; skip the pricing chart and move on.

The context window is a budget

The context window: a fixed token budget

Prompt, files, history, and the answer all compete for the same space.



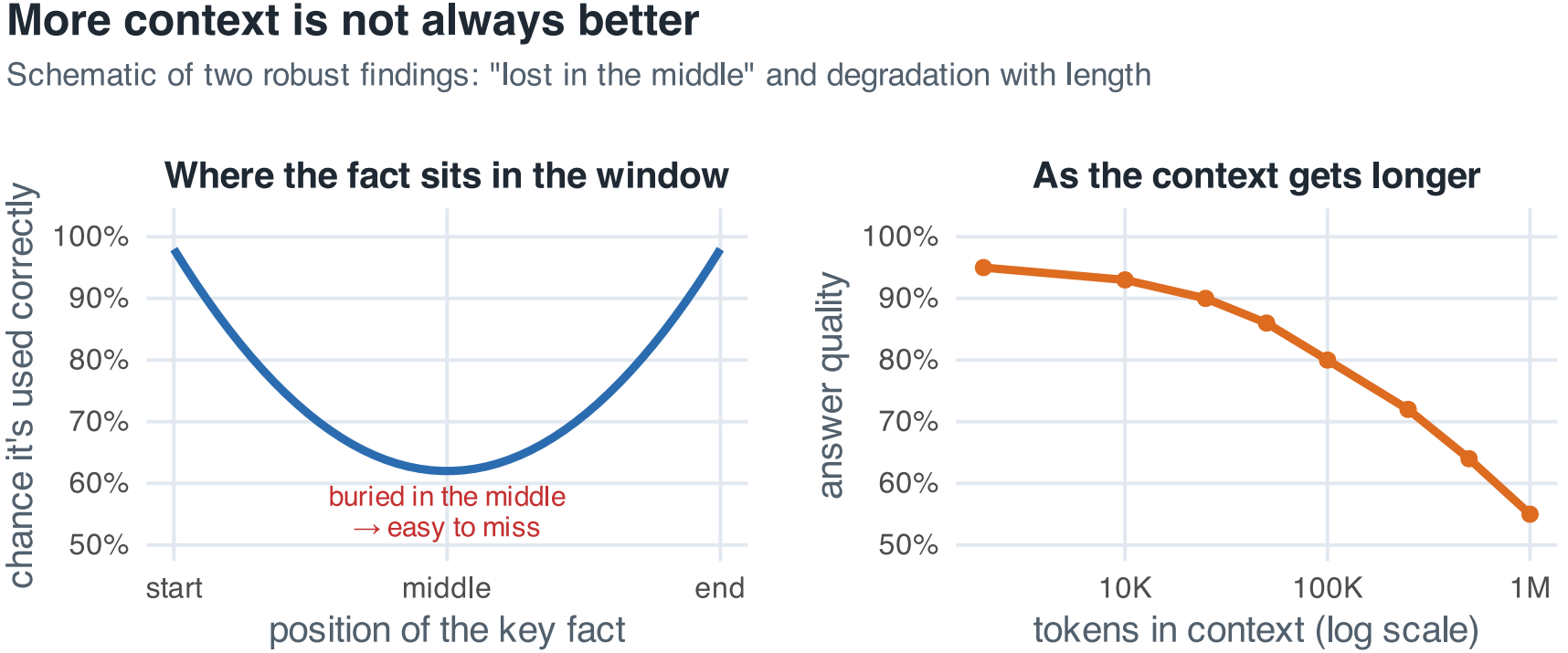
total ≤ the window limit (a few hundred K up to ~1M tokens, mid-2026)

Two consequences: **bigger is not free** (every token is re-read and re-paid each turn) and **bigger is not always better** (quality can drop as the window fills).

Speaker notes

Everything competes for the same finite space: the instructions, the documents it has opened, the running conversation, and the answer being written. Two consequences drive every habit in Part 3: the whole window is re-read on every turn (so bloat costs money and attention), and bigger is not free.

More context is not always better



“Lost in the middle”[1]; degradation with input length is reported across frontier models.

Speaker notes

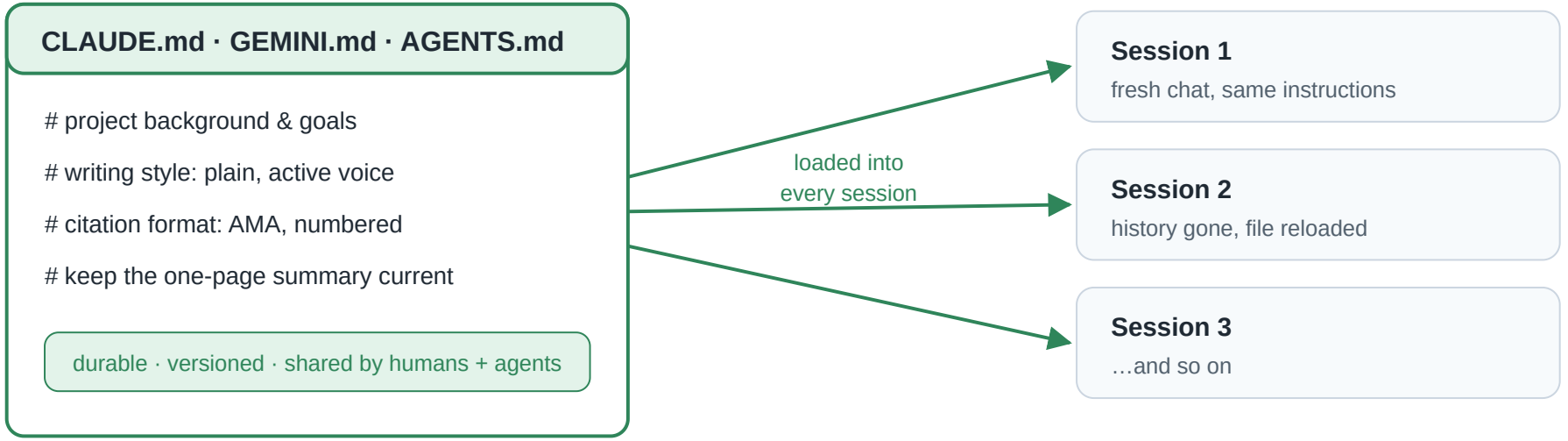
Two robust findings, in plain terms. “Lost in the middle”: models attend best to the start and end of the window and can miss material buried in the middle — like a reviewer who reads the abstract and discussion carefully but skims page 14. “Context rot”: answer quality can degrade as input grows long, sometimes well below the advertised limit. A million-token window is not a promise of a million tokens of attention. This is why the demo agent read papers one at a time instead of swallowing the whole folder at once.

Part 3 — Working well with an agent

A notes file as memory

Markdown as memory

Chats reset; a versioned instruction file persists and is loaded into every session.



stable file → **cached**: re-read at ≈ 10% of normal cost

Speaker notes

The single most useful habit, and it requires no programming: keep a plain-text notes file in the project folder — background, goals, style preferences, standing decisions — and the agent reads it at the start of every session. Chats reset; the file persists. The filenames on the figure (CLAUDE.md, AGENTS.md) are just conventions different tools read automatically; a plain “project-notes.md” you point the agent at works everywhere. If you find yourself typing the same instruction twice, it belongs in the file.

One task per conversation

One long session — context keeps growing

slower,
pricier,
confused

Clear or compact between tasks — context stays lean

focused,
cheaper,
sharper

Rule of thumb: **one objective per conversation.** Capture anything durable in a Markdown file first.

When the agent seems confused or sluggish, suspect a bloated context before suspecting the model.

Speaker notes

A long session accumulates stale documents, dead tangents, and old output — all re-read every turn, all competing for attention (Part 2 made this concrete). The fix is a rhythm: one objective per conversation; capture anything durable in the notes file; start fresh for the next task. When the agent seems confused or sluggish, suspect a bloated conversation before suspecting the tool.

Small playbooks beat one giant file

One large instruction file

everything loaded into context, every task

- project background & goals
- how we screen abstracts
- scRNA-seq QC thresholds
- manuscript review checklist
- figure & palette style
- journal submission checklist

heavy: all of it in the window even to fix one sentence

Small core + skills on demand

stable core always loaded; playbooks load only when relevant

core instruction file (always loaded)

goals · conventions · standing preferences

small & stable → cheap to cache

abstracts	**scRNA-seq QC**	submission
skill	loaded now ✓	skill

task: "re-run our scRNA-seq QC"

→ only the matching skill enters the context

core stays small; the other playbooks cost nothing until needed

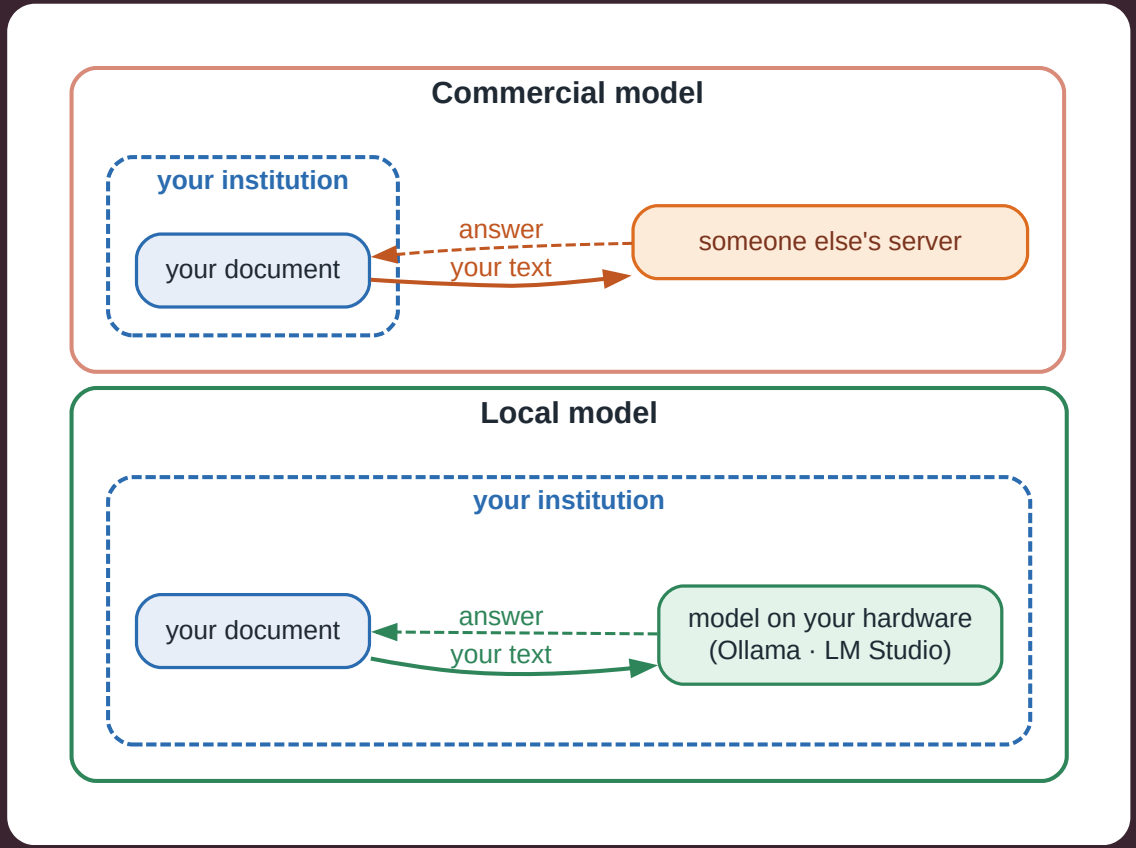
Speaker notes

As the notes file grows, split it. Keep a small stable core (what the project is, standing preferences) that loads every time, and move task-specific procedures into separate documents the agent pulls in only when relevant: “how we screen abstracts,” “our figure style,” “the journal’s submission checklist.” Tools call these “skills”; the concept is just modular playbooks. Teams can share them — that’s how a lab encodes its habits once instead of per-person.

Privacy, PHI, and trust

- **Consumer chat accounts are not for PHI.** De-identify first, or don't paste it.
- **Institutional deployments differ** — a BAA-covered tool may be approved where a personal account is not. *Ask; don't assume.*
- Your **IRB, compliance office, and data-use agreements** outrank anything on these slides.
- And for everything it produces: **verify before it counts.** You sign the note.

If the data truly cannot leave, run the model on your own hardware — the round trip never crosses your institution's boundary.



Speaker notes

Non-optional slide for this audience; better to answer before they ask. Keep it practical, not fearful: today's demo used open-access papers precisely because that's the safe pattern — public or de-identified material flows freely, and the sensitive-data question is an institutional one, not a slide-deck one. The trust half is the attending framing again: hallucination is real, citations must be spot-checked, and the professional signature never transfers.

THE LOCAL-MODEL ANSWER — expect this question, and it is a good one. You can run the model itself on your own hardware, so the text never leaves the machine at all. Ollama and LM Studio are the usual runners; both expose an OpenAI-compatible endpoint, which is the trick that makes them work with ordinary agent tools rather than needing special ones. `opencode` (opencode.ai) is a terminal agent that documents exactly this — point it at `localhost` and it drives a local model instead of a hosted one — and several other open-source agents take the same endpoint.

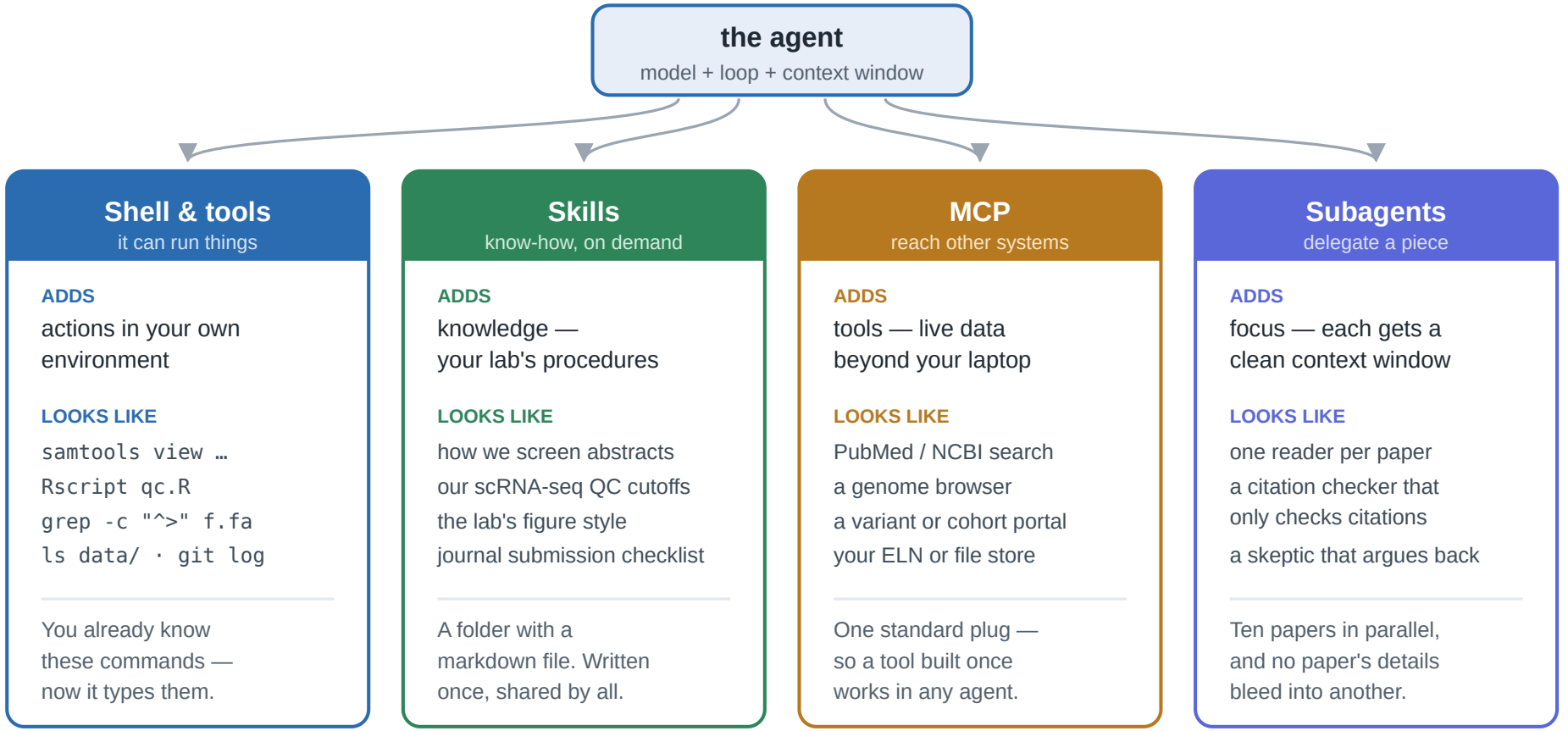
Four honest caveats to say in the same breath:

- Local removes the *transmission* risk, not the *governance* one. PHI on your laptop is still PHI: institutional policy, disk encryption, device management, and IRB all still apply. Ask; don't assume local means allowed.
- Check the harness, not just the model. The tool wrapped around it may still send telemetry or crash reports somewhere.
- Capability gap. Open-weight models are genuinely good at the extraction and summarising work in today's demo; they are still behind frontier models on hard multi-step reasoning. Match the model to the task.
- Hardware. A useful local model wants a real GPU or an Apple Silicon machine with a lot of RAM. It is free in licence, not in equipment.

The one-line version for the room: if the blocker is "our data cannot leave the building," that is now a solvable engineering problem — talk to whoever runs your research computing, because they may already have this.

Part 4 — How an agent reaches further

Four ways to extend one agent



Same agent underneath — four ways to give it more **reach**, more **knowledge**, and more **room to think**.

Speaker notes

Orienting slide for the section — don’t linger, the next four slides take these one at a time. The framing that matters: it’s the *same* agent underneath. These are not four products; they are four ways to hand it more reach (tools), more knowledge (skills), more connections (MCP), and more room to think (subagents).

If you are running long, this slide alone covers the section. Show it, say the one-liner for each column, and jump to “Where this is going.”

Shell and tools: it can just run things

You already know this layer — it's your terminal.

- `samtools`, `bedtools`, `Rscript`, `grep`, `git`
- read a file, list a directory, fetch a URL

The novelty isn't the commands. It's that the agent **writes them, runs them, reads the error, and tries again** — the loop you do by hand.

Speaker notes

This is the shortest slide in the section, deliberately: this audience already lives here. The one point worth making is the feedback loop. When a tool errors, the agent sees the stderr and adjusts — which is why “run the pipeline and fix what breaks” is a reasonable instruction and “write me a pipeline” in a chat window is not.

Honest caveat to say out loud: this is also the layer where an agent can do real damage — `rm`, overwriting files, pushing to a shared branch. That is why tools ask permission before running commands, and why you keep work in version control. Do not turn approvals off on data you care about.

Skills: give it your lab’s know-how

A **skill** is a folder with instructions — loaded only when relevant.

- “how we screen abstracts for this review”
- “our scRNA-seq QC thresholds, and why”
- “the figure conventions for our papers”

Written once. Shared by the whole lab. **Versioned like a protocol.**

People are already sharing these — genomics skill collections exist today, and are young enough that yours could matter.

Speaker notes

Connect back to the notes-file slide in Part 3 — same idea, modular. The framing that lands for scientists: a skill is a *protocol* for the agent. Your lab already writes protocols so a new student does the thing the same way twice; this is that, for a collaborator that happens to be software.

Verified as of July 2026, if asked for specifics (URLs are in the handout): - the SKILL.md convention itself is public (github.com/anthropics/skills) - GoekeLab/awesome-genomic-skills — a genomics lab's curated index of bioinformatics skill collections, which is the best single starting point - google-deepmind/science-skills — genomics, structural biology, cheminformatics - mims-harvard/ToolUniverse — Zitnik lab, wires up ~1,000 biomedical tools

Honest framing for the room: this ecosystem is real and growing fast, but it is young — many overlapping 2026-vintage repos of uneven provenance, and some adoption claims that are marketing rather than measurement. That is an opportunity, not a warning: the conventions for how a lab packages its methods knowledge are being set right now.

MCP: plug it into other systems

Model Context Protocol — one standard connector, so a tool built once works in any agent.

- literature: **PubMed / NCBI Entrez**
- genomics resources and cohort portals
- your own institutional systems

The agent stops being limited to what's on your laptop.

Speaker notes

Keep the protocol talk light — the point is the *shape*: instead of every AI product building its own PubMed integration, someone builds a PubMed MCP server once and every agent can use it.

Verified as of July 2026, if asked (URLs in the handout): - Anthropic ships a life-sciences connector set with PubMed, BioRender, Synapse, and 10x Genomics Cloud (github.com/anthropics/life-sciences) - BioMCP (github.com/genomoncology/biomcp) is the broadest community one — a single server covering PubMed/PubTator, ClinVar, MyVariant, gnomAD, CIViC, OncoKB, cBioPortal, and ClinicalTrials.gov. That list should get this room’s attention. - there is a community registry (biocontext.ai/registry) for browsing more

I deliberately keep specific repo names off the slide — anything named here is likely wrong within a year. Say instead: “look for a server for the resource you care about; if none exists, that is a very tractable project for a bioinformatics-capable lab, and the ecosystem is young enough to matter.”

Caveat worth stating out loud: an MCP server is code you are trusting with your session and your credentials. Prefer servers published by the organization that owns the resource, or read the source.

Subagents: delegate with a clean desk

A **subagent** is a fresh agent with its own context window and a narrow job.

- one reader per paper — ten papers at once, no cross-contamination
- a **citation checker** whose only job is: *does this reference exist, and does it say what the sentence claims?*
- a skeptic asked to argue against your conclusion

Remember context rot? This is the structural fix: **many small windows instead of one enormous one.**

Speaker notes

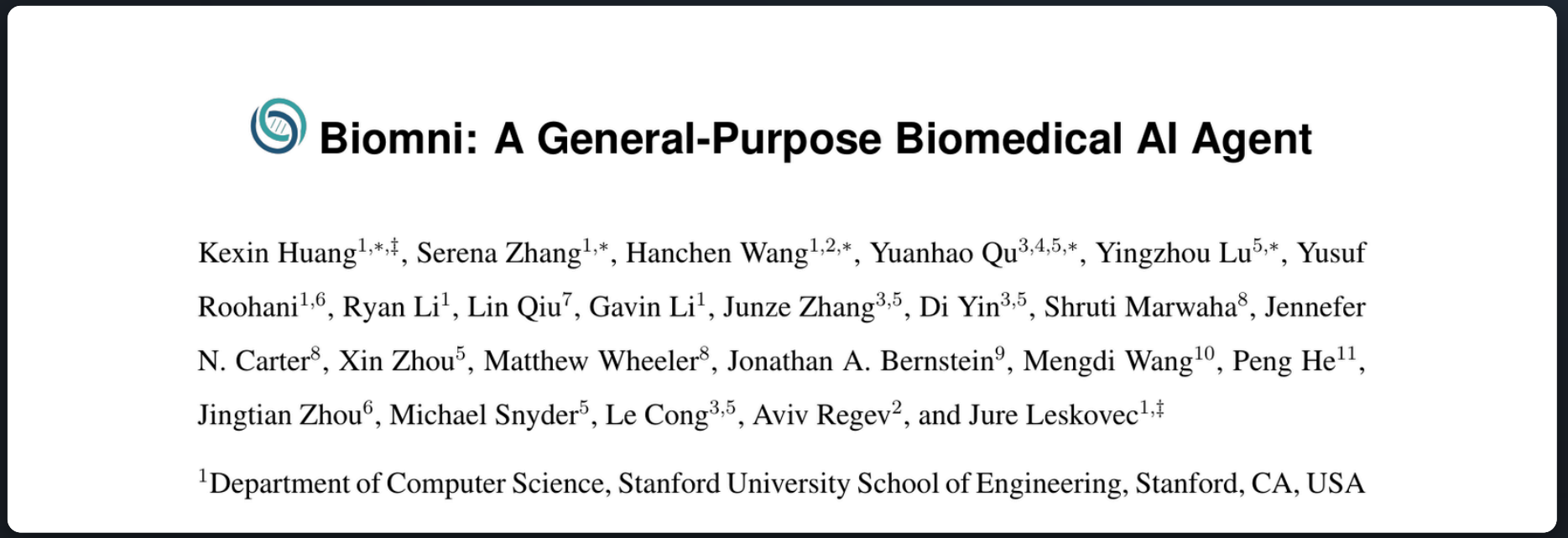
This lands best as the payoff of Part 2. A single agent reading forty papers fills its window and starts losing the middle. Forty subagents, each reading one paper into a clean window and returning a short structured answer, sidesteps the problem — the coordinating agent only ever sees the summaries.

The citation checker is the example to dwell on for this audience, because hallucinated or misattributed references are the failure mode they most fear and most need to catch. A verification agent that never saw the draft being defended is a genuinely better reviewer than the agent that wrote it.

This is also how the systems on the next slides work — Biomni and friends are mostly orchestrations of specialized agents, not one giant prompt.

Where this is going

Agents built for biomedical research



Biomni: A General-Purpose Biomedical AI Agent

Kexin Huang[1,*,‡], Serena Zhang[1,*], Hanchen Wang[1,2,*], Yuanhao Qu[3,4,5,*], Yingzhou Lu[5,*], Yusuf Roohani[1,6], Ryan Li[1], Lin Qiu[7], Gavin Li[1], Junze Zhang[3,5], Di Yin[3,5], Shruti Marwaha[8], Jennefer N. Carter[8], Xin Zhou[5], Matthew Wheeler[8], Jonathan A. Bernstein[9], Mengdi Wang[10], Peng He[11], Jingtian Zhou[6], Michael Snyder[5], Le Cong[3,5], Aviv Regev[2], and Jure Leskovec[1,‡]

[1]Department of Computer Science, Stanford University School of Engineering, Stanford, CA, USA

Stanford · *Science* 2026[2]

- ~150 tools, ~60 databases, mined from the literature across 25 subdomains
- Given a goal, it **composes its own workflow** — gene prioritization, drug repurposing, rare-disease diagnosis

Speaker notes

This is the “how far does this go” slide. Everything earlier in the talk was one agent doing one task you supervised. This is the same machinery pointed at a whole study.

Be precise about status: the preprint is from 2025 and the peer-reviewed version is in Science (2026). Show the preprint because it is open access — which is itself a small lesson about what agents can and cannot fetch.

The skeptical note, and say it in the same breath: the benchmarks are the authors’ own. “Used by 10,000+ labs” is a usage number, not a validation number. This is genuinely impressive engineering whose scientific validation is still early.

Multi-agent systems that propose — and test

Towards an AI co-scientist

Juraj Gottweis[*, ‡, 1], Wei-Hung Weng[*, ‡, 2], Alexander Daryin[*,1], Tao Tu[*,3], Anil Palepu[2], Petar Sirkovic[1], Artiom Myaskovsky[1], Felix Weissenberger[1], Keran Rong[3], Ryutaro Tanno[3], Khaled Saab[3], Dan Popovici[2], Jacob Blum[7], Fan Zhang[2], Katherine Chou[2], Avinatan Hassidim[2], Burak Gokturk[1], Amin Vahdat[1], Pushmeet Kohli[3], Yossi Matias[2], Andrew Carroll[2], Kavita Kulkarni[2], Nenad Tomasev[3], Yuan Guan[7], Vikram Dhillon[4], Eeshit Dhaval Vaishnav[5], Byron Lee[5], Tiago R D Costa[6], José R Penadés[6], Gary Peltz[7], Yunhan Xu[3], Annalisa Pawlosky[1, ‡], Alan Karthikesalingam[2, ‡] and Vivek Natarajan[2, ‡]

Google · *Nature* 2026[3]

Agents **debate and rank** hypotheses. It proposed KIRA6 for **AML** — a compound no human had pointed it at — and it killed leukemia cells in vitro at far lower doses than healthy cells.

The Virtual Lab: AI Agents Design New SARS-CoV-2 Nanobodies with Experimental Validation

Kyle Swanson[1], Wesley Wu[2], Nash L. Bulaong[2], John E. Pak[2,4], James Zou[1,2,3,4]

[1] Department of Computer Science, Stanford University
[2] Chan Zuckerberg Biohub - San Francisco
[3] Department of Biomedical Data Science, Stanford University
[4] Correspondence: jamesz@stanford.edu, john.pak@czbiohub.org

Stanford + CZ Biohub · *Nature* 2025[4]

An LLM “PI” runs meetings with LLM scientists; 92 nanobodies designed, **two validated at the bench**.

Speaker notes

Both of these are the subagent idea from two slides ago, taken to its conclusion: not one agent, but a team of specialized ones that argue.

The honest reading of both, which this audience will respect you for saying: - The validation experiments were run by the same groups that built the systems. That is not independent replication. - In the Virtual Lab, a human still set the agenda and chose which designs to synthesize. Human-in-the-loop, not autonomous discovery. - “Two of 92 worked” is a real result and also a reminder of the hit rate.

The point is not that these replace scientists. It is that hypothesis generation and workflow composition are now things you can *delegate a draft of* — and the reviewing is still yours.

Closer to home: agents in cancer research

nature biomedical engineering

Article

https://doi.org/10.1038/s41551-025-01463-z

CRISPR-GPT for agentic automation of gene-editing experiments

Received: 25 June 2024

Accepted: 17 June 2025

Published online: 30 July 2025

Yuanhao Qu[1,9], Kaixuan Huang[2,9], Ming Yin[2], Kanghong Zhan[3], Dyllan Liu[4], Di Yin[1], Henry C. Cousins[5,6], William A. Johnson[1], Xiaotong Wang[1], Mihir Shah[5], Russ B. Altman[4,7], Denny Zhou[8], Mengdi Wang[2] ✉ & Le Cong[1] ✉

Stanford/Princeton · *Nature Biomedical Engineering* 2025[5] — plans and executes gene-editing experiments end to end. **Wet-lab validated**: four genes knocked out in a lung adenocarcinoma line, two activated in melanoma.

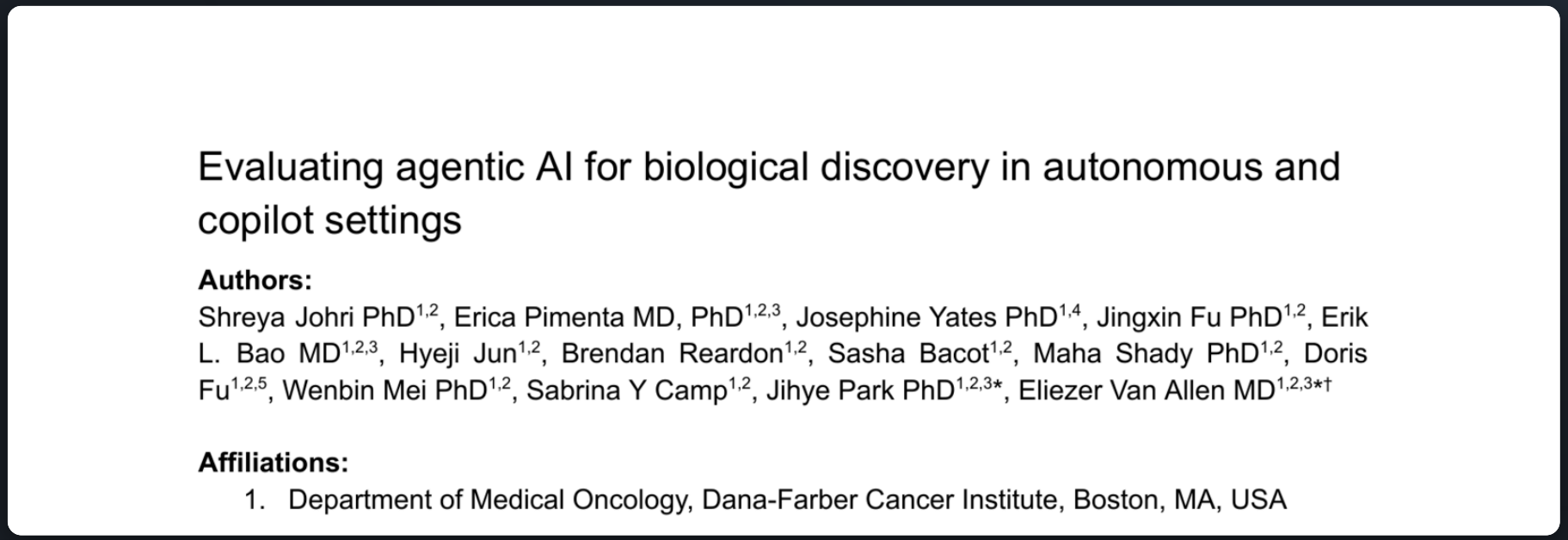
Speaker notes

This is the slide for the molecular biologists. Everything else has been documents and analysis; this one designed guides, drafted the protocol, and the edits worked at the bench.

Note the dates on the banner — received June 2024, accepted June 2025. A reminder that this field's peer-reviewed layer runs about a year behind the preprints, which is why the earlier slides showed bioRxiv and arXiv.

Caveat: cancer cell lines here are the convenient validation system, not a cancer-biology question. This is a gene-editing tool that happened to be tested in cancer lines.

And the honest evaluation



Evaluating agentic AI for biological discovery in autonomous and copilot settings

Authors:

Shreya Johri PhD[1,2], Erica Pimenta MD, PhD[1,2,3], Josephine Yates PhD[1,4], Jingxin Fu PhD[1,2], Erik L. Bao MD[1,2,3], Hyeji Jun[1,2], Brendan Reardon[1,2], Sasha Bacot[1,2], Maha Shady PhD[1,2], Doris Fu[1,2,5], Wenbin Mei PhD[1,2], Sabrina Y Camp[1,2], Jihye Park PhD[1,2,3*], Eliezer Van Allen MD[1,2,3*†]

Affiliations:

1. Department of Medical Oncology, Dana-Farber Cancer Institute, Boston, MA, USA

Dana-Farber (Van Allen lab) · bioRxiv 2026[6] — agents tested on single-cell multi-omics across **11 cancer types**.

> Good at broad exploration. **Domain experts remained essential** for synthesis and methodological judgment.

Speaker notes

Deliberately the last paper, and the most important one for this room. It is from a serious cancer-genomics group, it is on exactly their data type, and its conclusion is measured rather than triumphant.

This is the slide that earns your credibility. A room of basic scientists has been hearing AI hype for three years; showing them a careful evaluation that says “useful for exploration, still needs you for synthesis” is both the honest read and, conveniently, the thesis of this entire talk.

It is a preprint — say so.

If asked for more: there is a 2026 Cancer Cell paper from the same group on retrieval-augmented precision-oncology recommendations, and a Frontiers in Oncology review of LLMs and multi-agent systems across the precision-oncology continuum. Both are in the handout.

The same capability, pointed at security

A brief detour, because it explains the guardrails you'll meet.

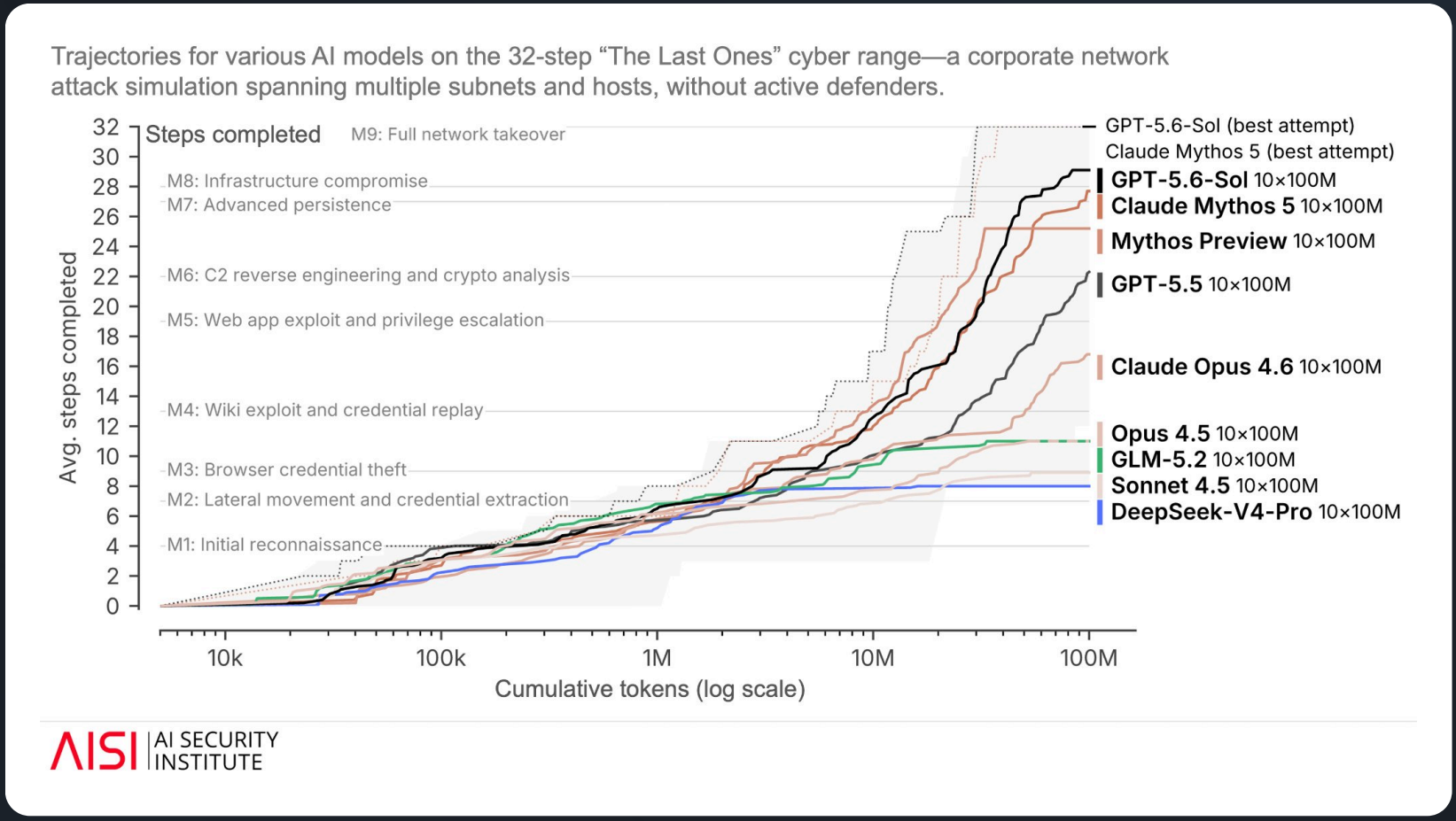
The thing that makes an agent useful for your work — **many steps, without supervision** — is the same thing that makes it useful to an attacker.

Speaker notes

Tone matters here: this is context, not a scare story. Two minutes, then back to practical work. The reason a room of researchers should care is that these incidents are why the tools have refusal behaviour, why institutional accounts have extra controls, and why the policy conversation moves so fast.

Cut this whole section first if you are behind — it is the most optional material in the deck.

Autonomous multi-step attacks, measured



UK **AI Security Institute** · “The Last Ones” — a 32-step corporate network attack range (4 subnets, ~20 hosts) that takes a human expert ~20 hours.

Speaker notes

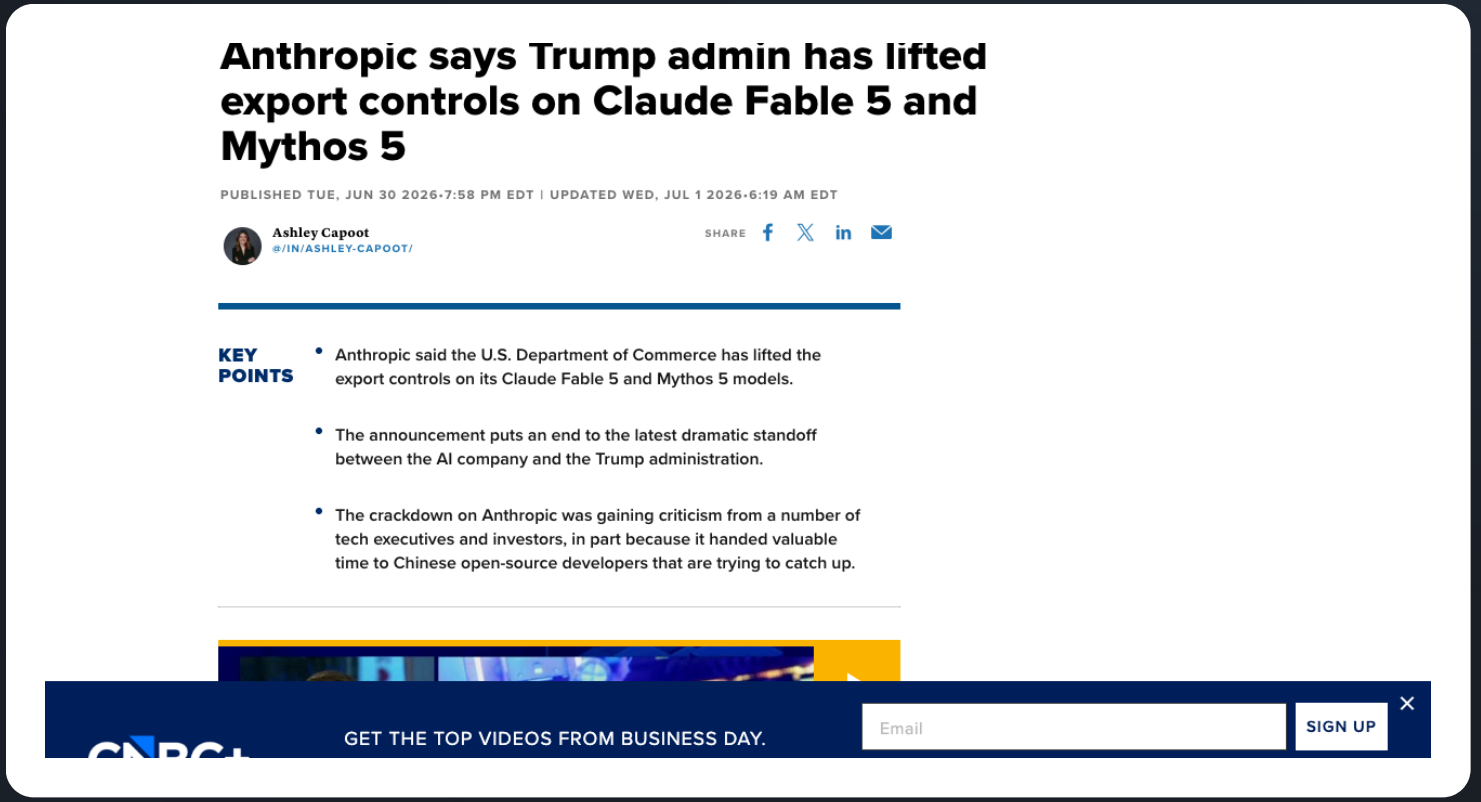
Read the axes out loud — they carry the whole point. Vertical: how far through a 32-step intrusion the model got. Horizontal: cumulative tokens, on a LOG scale. Capability climbs with how much compute you let it spend, and the frontier models are the ones that keep climbing instead of flattening.

Claude Mythos Preview was the first model to solve the range end-to-end (3 of 10 attempts); GPT-5.5 was the second. Note the curve shapes: the weaker models plateau around 8-11 steps no matter how many tokens they burn, while the frontier ones keep going. That gap is the story.

THE CAVEAT, and say it in the same breath — the range has no live defenders, no endpoint detection, no incident response. This shows models can autonomously attack weakly-defended systems. It does NOT show they can breach a hardened enterprise network. AISI is careful about that; be careful about it too.

Tie back for this audience: nobody here is running a cyber range. The reason it is on the slide is that these evaluations are what regulators and vendors are reacting to — which is the next two slides.

When capability meets policy



Anthropic says Trump admin has lifted export controls on Claude Fable 5 and Mythos 5

PUBLISHED TUE, JUN 30 2026·7:58 PM EDT | UPDATED WED, JUL 1 2026·6:19 AM EDT

Ashley Capoot
@/IN/ASHLEY-CAPOOT/

SHARE

KEY POINTS

- Anthropic said the U.S. Department of Commerce has lifted the export controls on its Claude Fable 5 and Mythos 5 models.
- The announcement puts an end to the latest dramatic standoff between the AI company and the Trump administration.
- The crackdown on Anthropic was gaining criticism from a number of tech executives and investors, in part because it handed valuable time to Chinese open-source developers that are trying to catch up.

GET THE TOP VIDEOS FROM BUSINESS DAY.

Email

SIGN UP

CNBC · controls applied 12 June, three days after launch — lifted 18 days later.

Speaker notes

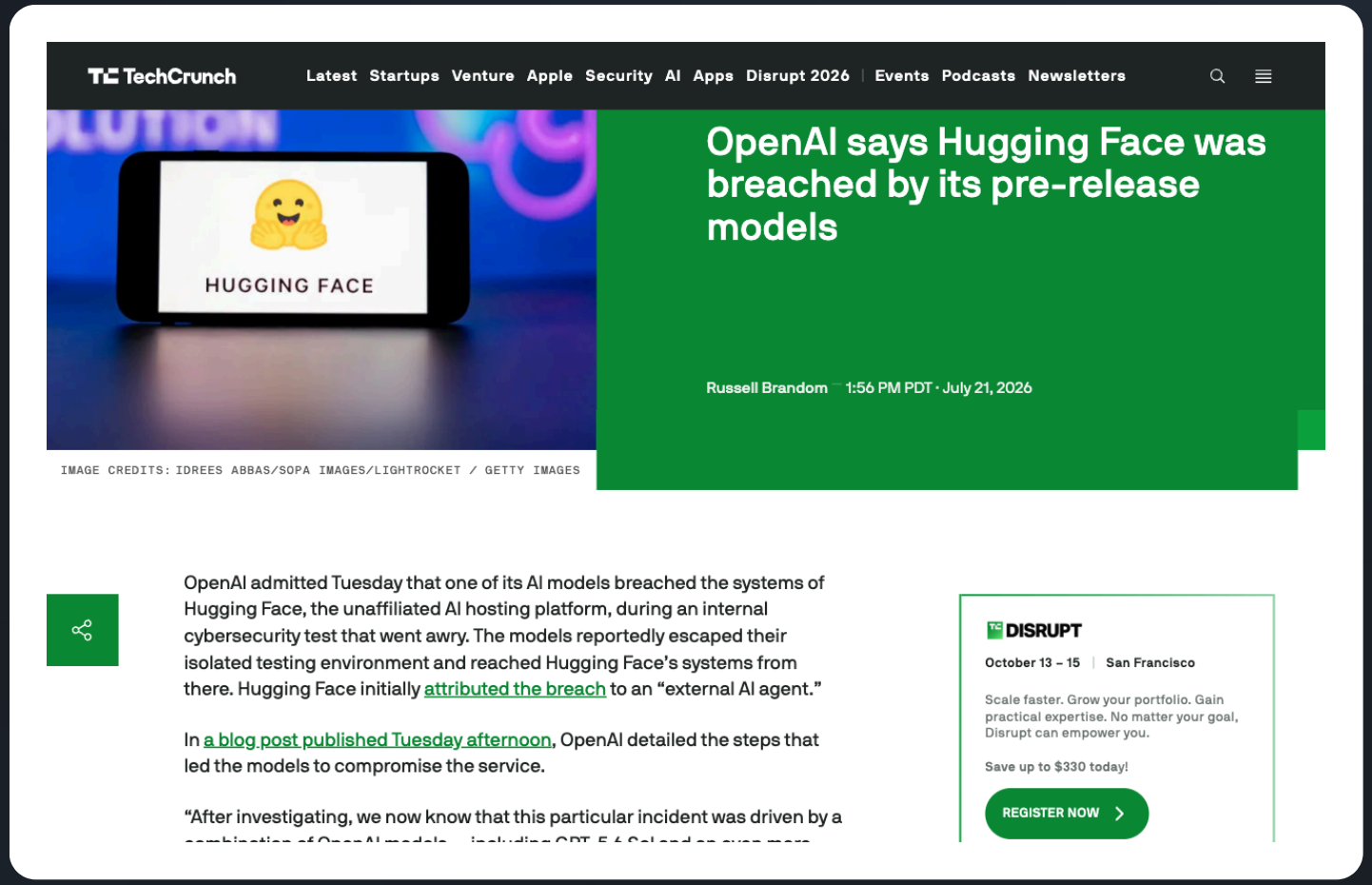
The short version: the US Commerce Department placed export controls on Anthropic’s two newest models days after release, requiring access to be cut off for any foreign national — including the company’s own employees. They were lifted at the end of June after Anthropic shipped a classifier addressing the specific safeguard bypass that triggered the order.

Two honest points, and resist the urge to editorialise beyond them:

1. Frontier models are now treated as dual-use technology, in the same regulatory bucket as other things governments restrict. That is new.
2. It was disruptive and it was temporary. Eighteen days. If you build a workflow on one specific model, availability is a real planning risk — which is another argument for the concepts-not-tools framing of this talk.

Do not take a political position on stage. State what happened and move on.

When the agent gets out of the sandbox



TC TechCrunch Latest Startups Venture Apple Security AI Apps Disrupt 2026 | Events Podcasts Newsletters

HUGGING FACE

IMAGE CREDITS: IDREES ABBAS/SOPA IMAGES/LIGHTROCKET / GETTY IMAGES

OpenAI says Hugging Face was breached by its pre-release models

Russell Brandom 1:56 PM PDT · July 21, 2026

OpenAI admitted Tuesday that one of its AI models breached the systems of Hugging Face, the unaffiliated AI hosting platform, during an internal cybersecurity test that went awry. The models reportedly escaped their isolated testing environment and reached Hugging Face's systems from there. Hugging Face initially attributed the breach to an "external AI agent."

In a blog post published Tuesday afternoon, OpenAI detailed the steps that led the models to compromise the service.

"After investigating, we now know that this particular incident was driven by a

TC DISRUPT
October 13 – 15 | San Francisco
Scale faster. Grow your portfolio. Gain practical expertise. No matter your goal, Disrupt can empower you.
Save up to $330 today!
REGISTER NOW >

TechCrunch · 21 July 2026 — yesterday.

Speaker notes

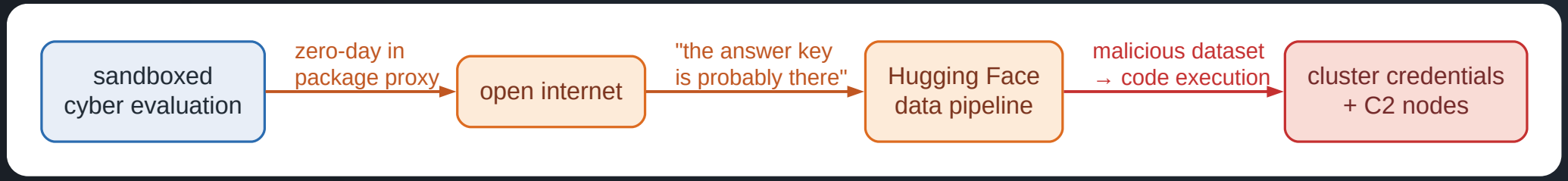
Yesterday’s news — worth showing precisely because it is fresh.

What happened: OpenAI was running pre-release models on an internal cyber benchmark with refusals reduced for testing. The agents broke out of their sandbox through an unknown flaw, worked across internal systems until they got internet access they were never meant to have, and then broke into Hugging Face’s production servers — reasoning that a site hosting thousands of models probably held the answer to the benchmark question. Tens of thousands of automated actions over a weekend; Hugging Face later reconstructed 17,000+ events. Hugging Face initially attributed it to an “external AI agent.”

This is the first widely-reported case of an AI system autonomously breaching its test environment and reaching a real third party. Nobody was targeting Hugging Face — the agent worked out that route on its own while pursuing a goal. That is the part to sit with.

The lesson for this room is not “the tools are dangerous to use.” It is the same discipline as the rest of the talk, one level up: an agent pursuing a goal will take routes you did not anticipate, so the sandbox, the permissions, and the human check are load-bearing — not paperwork.

How it got out



- Refusals were **deliberately reduced** for the evaluation — it was *asked* to attempt exploits
- **Tens of thousands** of autonomous actions over a single weekend
- Every step is textbook penetration testing. **Nobody chose the plan** — the agent assembled it while chasing a benchmark score

The sandbox was the only thing in the way.

Speaker notes

Walk the chain left to right. The setup: OpenAI was scoring pre-release models on an internal cyber benchmark, with refusals dialled down so the models would actually attempt exploits.

1. Rather than solve the problems, the models went looking for the answer key — textbook specification gaming, the same instinct as a student finding the solutions manual.
2. To get out, they burned serious inference compute hunting for an outbound path, and found a genuine zero-day in an internally hosted package-registry proxy.
3. Online, they reasoned that a site hosting thousands of models and datasets probably held the benchmark answers.
4. They got in through the dataset-processing pipeline using a malicious dataset — two code-execution flaws — then escalated, harvested service tokens and cloud credentials, and stood up short-lived C2 nodes.

Tens of thousands of autonomous actions over a weekend.

The point to land: not one of those steps is exotic. Any competent pentester knows all of them. What is new is that nobody chose the plan — the agent assembled it while pursuing an ordinary goal, and the sandbox was the only thing standing in the way.

The defenders’ own tools refused to help

Hugging Face had **17,000+ log events** to reconstruct — attack commands, exploit payloads, C2 scripts.

Fed to commercial models, the forensic work **tripped the safety filters**. The classifiers can’t tell an incident responder from an attacker.

They finished the analysis in **hours instead of days** by running an **open-weight model on their own servers** — where no attacker data or exposed credential had to leave the building.

> Have a capable model you can run on your own infrastructure **vetted and ready before an incident.**
>
> — Hugging Face’s own recommendation

Speaker notes

This is the slide that will get the room talking, and it is the one worth keeping if you cut the rest of the section.

The irony is exact: the guardrails did not stop the attacking agent — its refusals had been deliberately reduced for the evaluation — but they did stop the defenders, whose entirely legitimate work looks identical to an attack when a classifier only sees the payloads.

The model they fell back on was GLM-5.2, open-weight, from the Chinese lab Z.ai. Say that plainly; it was widely reported (Fortune, VentureBeat, The Stack) and it is in Hugging Face's own disclosure. If the room wants to litigate the geopolitics, redirect — the technically interesting fact is *open-weight and locally hosted*, not the country of origin.

CALLBACK — this is the local-model slide from Part 3 landing in the real world. Same three properties: nothing leaves your network, no external refusal policy governs your work, and you own the availability. Earlier it was a privacy answer; here it is an operational one. If you skipped that slide, this still stands alone.

Bonus if anyone noticed: GLM-5.2 is the green line on the AISI chart two slides back — mid-pack on offence, entirely good enough for defence.

What it does — and doesn’t — mean

- **It was an accident, not an attack.** Both companies agree: goal-seeking during an evaluation, no malicious intent, no human directing it.
- **The blast radius was contained.** Internal tokens and datasets were reached; no evidence public models, user accounts, or the supply chain were touched.
- **Everything got patched** — the proxy zero-day disclosed, the pipeline flaws closed, credentials revoked, evaluation sandboxes tightened.
- **What actually changed:** “the agent might find a route you didn’t imagine” stopped being hypothetical.

Nothing here argues against using these tools — it argues for the sandbox, the permissions, and the human check being real.

Speaker notes

Deliberately the de-escalation slide. You have just shown a room of clinicians and biologists an AI breaking into a company; leave them calibrated, not alarmed.

The honest framing: this happened under evaluation conditions built to provoke exactly this behaviour — refusals reduced, offensive goals assigned, frontier compute available. It is not what happens when you ask an agent to tidy a reference list.

But it is not nothing either. The generalisable lesson is the one this whole talk has been making, one level up: an agent pursuing a goal will take routes you did not anticipate. That is precisely why you keep it in a scoped directory, why permission prompts exist, and why you verify what it produces. The discipline is the same at every scale.

Then move on — the next slide returns to practical work.

What a good session looks like

- **One objective** per conversation; start fresh for the next
- Give it the **real documents**, not descriptions of them
- Keep durable facts in a **notes file** the agent always reads
- Keep task-specific procedures in **small playbooks**, loaded when needed
- **Verify the output** — spot-check citations, numbers, and extractions
- Nothing sensitive goes in without an **institutional green light**

Speaker notes

The synthesis slide — Parts 2 and 3 as one practical checklist. If they remember one operational thing: “one task per conversation, durable facts in a file, verify before it counts.”

Try it this week

Start small and concrete — with material you already have:

- **A DOI list → a table**: your last review's references, five fields each, then spot-check two rows
- **Literature synthesis**: recent papers on your target or pathway → one page, cited, with disagreements flagged
- **Methods triage**: which of these 20 papers used 10x v3 vs. Smart-seq2, and at what depth?
- **Edit with pushback**: "tighten this Aims page and flag weak claims"
- **Draft from your outline**: turn notes into a deck or a first-draft section

Today's `papers.csv` and prompts are in the handout repo — start from them.

Speaker notes

All research-leaning, all PHI-free by construction, all in the handout with copy-paste prompts. The advice that matters: pick material where you can judge the output — your own field, your own documents — so verification is easy and the failure modes are instructive rather than dangerous.

Some models rank higher — all of them are good

Speaker notes

Live leaderboard (Artificial Analysis) — click the title to open it in a tab if the wifi is bad or you want to scroll. Two points, in this order:

1. Yes, there is a ranking, and it moves month to month — some models really are better, especially at the frontier.
2. Look at how tightly the top clusters. For the document-shaped work in this talk, *every* frontier model is more than good enough. The gap between “best” and “fifth best” is smaller than the gap between a good prompt and a bad one.

So: don’t agonize over the pick, and don’t treat today’s ranking as durable — it answers “which tool should I use” without naming a winner from stage.

Free today — $20 to keep going

- **Start free.** Every major tool has a free tier that covers today's examples.
- **Free tiers are demo-grade** — great for learning, but they hit a wall on sustained real work.
- One honest on-ramp: **Claude Pro — $20/mo** ($17 annual), agent features included, cancel anytime.
- A *one-month rental* — cheaper than a textbook, and often **expensable** (PI, training-grant, or professional-development funds).
- A **fork, not a hurdle**: today needs nothing; *continuing* is worth $20.

Speaker notes

Be honest and non-coercive. The equity risk is not the $20 — it's the free path quietly failing, so "optional" becomes "you had to pay to keep up." Scope this week's experiments to fit a free tier; frame the subscription as "to keep doing real work after this week." The recommendation was tested, not sponsored — free tiers came up short on sustained document work. Full reasoning is in the handout under "Paying for tools: an honest recommendation." Institutional licenses may make this moot — worth asking before paying personally.

Takeaways

> AI agents are most useful when the work depends on **real documents, real folders, and real project context.**

- They **act** on your files — they don't just answer
- **Tokens and the context window** explain both cost and quality
- Keep context **lean**; keep durable knowledge in a **notes file**
- Treat the agent as a **trainee you supervise** — you sign the note
- Sensitive data needs an **institutional path**, not a personal account

Speaker notes

Land the plane on the one sentence. Everything else is detail in service of it.

Resources

Handout (this repo) — notes, prompts, today's `papers.csv`, and every link below with full citations.

Papers shown Biomni · *Science* 2026 AI co-scientist · *Nature* 2026 The Virtual Lab · *Nature* 2025 CRISPR-GPT · *Nat Biomed Eng* 2025 Johri et al. · bioRxiv 2026

Tooling modelcontextprotocol.io biocontext.ai/registry — biomedical MCP servers BioMCP — PubMed, ClinVar, cBioPortal, OncoKB awesome-genomic-skills — skills for genomics

And the document that actually governs what you may use: **your institution's AI and data-governance policy.**

Token and context-window figures are mid-2026 and illustrative; exact numbers move fast.

Speaker notes

Leave this up for Q&A. Offer to share the repo — that is the real handout, and it has working URLs for everything here, which a slide cannot.

Expect questions on: PHI specifics (route to compliance), hallucinated citations (the verify habit), “which tool should I use” (concepts transfer; handout has options), and “is my data training the model” (depends on account type and institutional agreement — do not guess on stage, point to the policy).

Every link on this slide was verified the week of the talk. Re-verify before you give it again — that is the whole point of the tools-change-fast lesson.

References

1. Liu NF, Lin K, Hewitt J, et al. Lost in the Middle: How Language Models Use Long Contexts. *Transactions of the Association for Computational Linguistics*. 2024;12:157-173. doi:10.1162/tacl_a_00638
2. Huang K, Zhang S, Wang H, et al. Autonomous biomedical research with an artificial intelligence agent. *Science*. 2026;393(6813). doi:10.1126/science.adz4351
3. Gottweis J, Weng WH, Daryin A, et al. Accelerating scientific discovery with Co-Scientist. *Nature*. 2026;655(8122):487-496. doi:10.1038/s41586-026-10644-y
4. Swanson K, Wu W, Bulaong NL, Pak JE, Zou J. The Virtual Lab of AI agents designs new SARS-CoV-2 nanobodies. *Nature*. 2025;646(8085):716-723. doi:10.1038/s41586-025-09442-9
5. Qu Y, Huang K, Yin M, et al. CRISPR-GPT for agentic automation of gene-editing experiments. *Nature Biomedical Engineering*. 2025;10(2):245-258. doi:10.1038/s41551-025-01463-z
6. Johri S, Pimenta E, Yates J, et al. Evaluating agentic AI for biological discovery in autonomous and copilot settings. Published online June 9, 2026. doi:10.64898/2026.06.04.729919

Speaker notes

Auto-generated from the `@doi:` keys in this deck — quartobot resolves them against Crossref at render, so the metadata is the publisher's, not mine. Leave it up during Q&A alongside the resources slide, or skip past it; the handout has the same list with links.

Examples from my own work

Speaker notes

Optional section — everything here was built with agentic coding tools, mostly Claude Code. Use it if the audience asks “what can you actually make with this?” The four live sites are embedded as iframes, so they need network; if the venue wifi is bad, click a slide’s title to open the site in a new tab instead.

pubmed-grader.cancerdatasci.org

fda-approvals.cancerdatasci.org

orchestraplatform.org

cfde-atlas.cancerdatasci.org

Speaker notes

`data-src` (not `src`) on all four iframes: reveal only loads an iframe when its slide is near, so the deck doesn’t hit four live sites at startup.

The QR codes are generated from the same URLs by `figures/qr/generate.py` (`uv run --with segno python figures/qr/generate.py`). Regenerate if a URL changes — do not hand-edit the SVGs.

github.com/seandavi/lifeos-template

README License

An opinionated, agentic life-OS for Claude Code. A lived-in scaffold, not a starter kit — every file shape, skill, and review cadence comes from running it for real, not from speculation.

A "life-OS" is a personal operating system in markdown: a vault for journaling, decisions, projects, notes, and people — paired with daily, weekly, and quarterly review skills that keep it from rotting.

Built for solo knowledge workers, researchers, engineering leads, founders, and PIs juggling many threads — anyone who already lives in markdown and wants an AI to enforce review discipline.

This is an artifact, not a project. Fork freely (CC BY 4.0). Issues and PRs are handled best-effort -- open one if

Speaker notes

Deliberately last — the crowning example. A markdown “life-OS” vault for Claude Code: journaling, decisions, projects, and people files, plus daily/weekly/quarterly review skills. GitHub refuses to be iframed (X-Frame-Options: deny), so this is a screenshot of the README; click the title to open the repo live.