Twenty Billion Parameters, One Big Harness

2026-06-07 / 00:16:51

> *"The capability moved out of the model this week. The judgment about where to trust it didn't move at all."*
>
> — from this episode's transcript

- Lenar Kess
- Damra Vol

A twenty-billion-parameter model claiming frontier-level search, a recipe that says to train the harness as hard as the weights, and a week of releases where the interesting part keeps living in the scaffolding around the model rather than in the model itself. Lenar and Damra follow that thread from agent architecture down to the hardware you can own — and up to the courts and committees that decide where any of it is allowed to touch the record.

- Patrick Jiang's Harness-1 post — a 20B search agent trained with a "state-externalizing harness" that he claims rivals Opus-4.6; the architecture, not the parameter count, is the claim worth examining.
- Viv's "agent = model + harness" recipe — train both components together; the same specialization logic shows up everywhere this week.
- Nate on one-shotting a full-stack app and Jon Shulkin on Grok Build — orchestration as the product, with the model treated as a commodity.
- CRUX's agent publishing an iOS app — "a few human interventions" is the detail that decides whether open-world evals beat pass/fail scores.

- Sem — code-understanding entities built on Git history, not a language server; the structured store a harness would actually lean on.
- Universal Memory Protocol vs Databricks' end-to-end Instructed Retriever — standardize memory, or specialize retrieval for a 3x win? The incentives point opposite ways.
- NVIDIA's RTX Spark at Korea's PC Bangs and the GLM Air/GGUF thread — the local crowd wants the smallest good-enough model on hardware they own.
- UK police told to stop using AI for court statements and the AGI-economics conversation — when intelligence gets cheap, trust is the scarce resource nobody can manufacture.

SEGMENTS

Transcript

1. Lenar Kess 00:00:04

   Yesterday afternoon, Patrick Jiang put up four lines about a model called Harness-1. It's a search agent with twenty billion parameters, and the claim is that on long-horizon search it rivals Opus-4.6, it beats GPT-5.4, and it runs at what he calls Context-1-level cost and latency. The line that made me stop wasn't the benchmark boast. It was how he described the training — a state-externalizing harness. So here's what I keep circling. When a twenty-billion-parameter model goes

step for step with a frontier model on a hard task, where did the capability actually come from — the weights, or the scaffolding wrapped around them?

- x.com
- x.com
- x.com
- x.com
- x.com
- ataraxy-labs.github.io
- v.redd.it
- pidgin.sh
- universalmemoryprotocol.io
- x.com
- blogs.nvidia.com
- reddit.com
- x.com
- reddit.com
- techmeme.com
- techmeme.com
- fortune.com

2. Damra Vol 00:00:43

   [tsk] Before either of us gets excited — that's a tweet, not a paper. Four lines, no eval card, no methodology. 'Rivals Opus-4.6 on long-horizon search' is the kind of sentence that lives or dies on which search benchmark you chose and how you scored a single run as a success. So I'm holding the number at arm's length. The architecture phrase is the part I can actually reason about, though. Let me be precise about what externalizing state means, because people throw that phrase around loosely.

3. Lenar Kess 00:01:15

   Do that, because I think it's the whole game here.

4. Damra Vol 00:01:18

   So a normal agent carries its working memory inside the model's context window — the search tree it has explored, the dead ends, the notes to itself, all of it sitting in tokens the model re-reads every step. That gets expensive fast, and it gets unreliable past a few dozen steps, because the model starts losing the early context. A state-externalizing harness pulls that working memory out of the window and into a store the harness manages. The model reasons over a small, curated slice each turn, and

the harness decides what to write down, what to fetch back, and what to throw away. So a small model can run a long search, because it's never holding the whole problem in its head at once.

5. Lenar Kess 00:02:00

   Which is the same recipe Viv was describing this morning, separately. She posted what she called a default recipe everyone should use — agent equals model plus harness, and you train both. Build a version-one agent on a sensible base harness with some task-specific data, then keep tuning the harness and the model together, instead of freezing one and pushing only on the other.

6. Damra Vol 00:02:22

   And that's where this field actually is right now. The interesting work isn't a bigger base model. It's the loop around the model: what state it keeps, what it externalizes, what the harness retries when a step fails. Harness-1, if the claim holds, is a proof that you can move a lot of that burden out of the weights. That matters, because twenty billion parameters runs on hardware a normal team can afford, and two hundred billion doesn't.

7. Lenar Kess 00:02:49

   We came back to this same point yesterday — the whole episode was about the harness carrying the model. Command Code's deterministic repair layer, the skill files covering what the weights couldn't. Two days running, the releases people are reacting to are about the harness around the model, not the parameter count.

8. Damra Vol 00:03:06

   Right, and there's a cost to that I want named. If the capability lives in the harness, the capability isn't portable. You can't take Harness-1's weights, drop them into your own loop, and get the same result — you'd be importing twenty billion parameters and leaving the actual intelligence behind. So the open question for me is whether Jiang releases the harness, or just the weights, or neither. That decides whether this is a contribution or a demo reel.

9. Lenar Kess 00:03:33

   Let's go to the place where this stops being theory. This morning Nate — he posts as natebirdman — said he's been gluing some DeepSeek agents together with a couple of tools he's building, and he can now one-shot a full-stack app, web plus iOS plus Android, for about a dollar, in twenty minutes.

10. Damra Vol 00:03:50

    One-shot is hiding a lot in that sentence. [chuckle] What does one-shotting a full-stack app actually produce? A scaffold that compiles? A thing with authentication, a database, and a deploy target?

Those are very different artifacts, and the distance between them is where almost every one of these demos falls apart.

11. Lenar Kess 00:04:10

    He doesn't say, and that's fair to flag. But notice the shape of the claim — it isn't about the model, it's about the orchestration. DeepSeek agents, plural, plus his own tooling. The model is a commodity in his story. The product is the glue between them.

12. Damra Vol 00:04:25

    Which rhymes with the Grok Build one. Jon Shulkin posted that Grok Build let him add a natural-language code and interface comment tool that's live inside the app being built — so he leaves a comment in plain English, and Grok Build makes the change and updates the running app. That's a tight loop. What makes it matter, if it holds up, is that the edit target is the deployed app, not a local sandbox you still have to ship by hand afterward.

13. Lenar Kess 00:04:53

    And then the most concrete of the three — CRUX tested an agent on building and publishing an iOS app all the way to the App Store. Tamaz Gadaev's read was that it worked with a few human interventions, and that open-world evaluations reveal more than a pass-or-fail score.

14. Damra Vol 00:05:09

    A few human interventions is the detail that matters in that whole cluster. Publishing to the App Store means provisioning profiles, signing certificates, review guidelines, screenshots, and a privacy manifest. If an agent got through all of that with a human stepping in a few times, the interesting question isn't did it pass. It's where exactly it needed a human, and whether that's the same place every single time. Because that's the part you would have to staff.

15. Lenar Kess 00:05:36

    That's the argument for open-world evaluations over benchmarks, though. A pass-or-fail score tells you the agent cleared a gate someone designed. Watching it stumble through App Store review tells you which gates actually matter in the real workflow.

16. Damra Vol 00:05:50

    Agreed, with one caveat — an open-world demo is also unfalsifiable marketing if nobody publishes the intervention log. 'It worked with a few human interventions' and 'it failed constantly and we rescued it' can be the same video, edited two different ways. So I'd want the transcript. Same thing I

want from Harness-1. The day someone ships one of these with the failure trace attached is the day I stop hedging.

17. Lenar Kess 00:06:17

    The agents get the attention. The tooling underneath them is what I find more durable. There's a project called Sem that showed up on Hacker News — a hundred and twenty-eight points, a real comment thread — pitched as a new primitive for code understanding. Not a language server, the author says. Entities built on top of Git.

18. Damra Vol 00:06:35

    And the demo command is what sells it. You run sem impact, give it a function name like authenticate-user, and it tells you that function depends on the database find-user call and the rate-limiter check, and it's used by the login route and the auth middleware. So it's a dependency graph of your code as semantic entities — not a text search, and not the language-server protocol's symbol table.

19. Lenar Kess 00:07:00

    Why does the Git part matter? The author makes a point of saying it's built on Git, not on a language server.

20. Damra Vol 00:07:06

    Because a language server understands your code at one moment — the current state of the files. Building the entities on Git means the graph has history. You can ask not just what depends on this function, but what depended on it three commits ago and what changed since. For an agent, that's the difference between a snapshot and a memory. If an agent is about to refactor that authenticate-user function, the impact set is exactly the context it needs, scoped down — not the whole repository dumped into the window.

21. Lenar Kess 00:07:37

    Which connects straight back to the externalized-state idea. This is the kind of structured store a harness would lean on instead of stuffing raw files into context.

22. Damra Vol 00:07:46

    I'd draw that connection too. And the same Hacker News crowd had the right reflex on a neighbor of this — the Universal Memory Protocol post. Someone pitched a shared format for agent memory, and the top comment was, essentially, this is only as good as adoption, so who is actually using it. That's the correct first question for any protocol.

23. Lenar Kess 00:08:06

    Hold that — memory deserves its own segment in a minute. Stay on the building tools for one more beat. There were two more I went through — a Python library for building Claude-Code-style terminal interfaces, and pidgin.sh, which lets Claude Code share an artifact as a link instead of you saving the file and finding somewhere to host it.

24. Damra Vol 00:08:25

    Both of those are small, and I mean that as a compliment. The terminal-interface library is someone noticing that half the work of a coding agent is the transcript view — messages streaming in, tool calls flipping from running to done in place — and packaging it so you don't rebuild it every time. pidgin is one friction point removed. Neither is a turning point. They're the connective tissue that makes the flashy demos usable day to day, and that's what most good weeks in this field actually look like.

25. Lenar Kess 00:08:55

    Let's take memory head-on, because two items today are the same problem from opposite ends. One is the Universal Memory Protocol — a proposed shared format so an agent's memory is portable across tools. The other is Matei Zaharia posting that Databricks made their Knowledge Assistant three times faster with a new Instructed Retriever model, trained end-to-end to do the retrieval.

26. Damra Vol 00:09:19

    And the tension between those two is the whole story of this space. The Universal Memory Protocol is betting that memory should be a standard — a neutral format everyone agrees on, the way the Model Context Protocol became the standard for tools. The Databricks result is betting the opposite. They didn't adopt a shared retriever. They trained their own, end to end, for their own assistant, and got three times the speed out of it.

27. Lenar Kess 00:09:44

    Is that a real contradiction, though? Standard format for storage, custom model for retrieval — those two can coexist in one system.

28. Damra Vol 00:09:52

    They can, and that's probably the right architecture. But watch the incentives. The Model Context Protocol caught on because Anthropic shipped it, and then a lot of tools wanted to talk to Claude, so there was a reason to adopt it: the thing on the other end of the handshake was valuable. A memory protocol has no equivalent magnet yet. The commenter nailed it — who is using this? Without one

big consumer everyone wants to interoperate with, a shared memory format is a good idea with nobody on the other side.

29. Lenar Kess 00:10:21

    And the Databricks side cuts the other way. If the performance win comes from training retrieval specifically for your data and your assistant, the incentive is to specialize, not to standardize. Three times faster is a number a product team will defend against any abstraction that costs them a slice of it.

30. Damra Vol 00:10:38

    I keep coming back to that. Every standard in this field is fighting the fact that the best results come from end-to-end specialization. Matei's retriever is trained inside the loop, just like the Harness-1 story and Viv's recipe. The pattern across the entire day is the same: train the component for your own system. A portable memory protocol is asking people to do the opposite, right when specialization is paying off. I'm not saying the protocol is wrong. I'm saying it's swimming upstream, and the commenters smelled it.

31. Lenar Kess 00:11:09

    Let's bring this down to the hardware, because none of it runs on faith. NVIDIA announced something called RTX Spark — they're calling it a superchip that reinvents the Windows PC — and the launch venue is the part I didn't expect. They rolled it out with KRAFTON, NC, and the T1 League of Legends team at Korea's PC Bangs, the gaming cafés.

32. Damra Vol 00:11:30

    Which is a sharp distribution choice, and the reasoning is the point. A PC Bang is a room full of high-end machines that thousands of people sit down at. If you want consumers to have a personal AI agent running on local hardware, seeding the gaming cafés puts the chip in front of exactly the people who will push it hardest. Calling it a personal agent on a superchip is marketing, but the hardware question underneath is concrete: can you run a capable agent locally, on a machine you own?

33. Lenar Kess 00:11:59

    And the answer from the local-model crowd this week is sort of, and it's frustrating. There's a thread on the local-models subreddit — the title is basically, Z.ai, we need Air, where's the GLM build in GGUF format. They're complaining that GLM 5.1 is a strong coding model that's too big for most people to run locally, and slow even on the API. They want the smaller Air variant back, and it hasn't come.

34. Damra Vol 00:12:25

    That's the exact gap Harness-1 is aiming at, if you connect the two. The local crowd doesn't want the biggest model. They want the smallest model that's still good enough, in a format that runs on their own hardware, because the whole point of local is that you own the loop. A twenty-billion-parameter agent that punches above its weight because of the harness is precisely what that thread is asking for — they just want it open and quantized.

35. Lenar Kess 00:12:50

    Two smaller hardware-adjacent items to close this out. mlx-audio shipped version 0.4.4, with new text-to-speech and speech-recognition models running locally on Apple Silicon. And someone posted a benchmarking tool called LLM API Benchy on the local subreddit, built because they couldn't find a benchmark that stayed consistent across inference engines.

36. Damra Vol 00:13:13

    The benchmark one is the more useful of the two, and it's the least glamorous. Everyone running local models has hit the problem that you can't compare two inference engines fairly, because the test keeps drifting between runs. A consistent harness for measuring tokens-per-second and latency across engines is the kind of shared yardstick that makes everyone else's numbers trustworthy. mlx-audio is just nice to have: more of the stack runs on a laptop now, with no API key and no round trip to a server.

37. Lenar Kess 00:13:44

    Let's end where the machines meet the institutions, because two stories this week put the limits in sharp relief. The first one is blunt: several UK police forces have been told to stop using AI to prepare court statements. The Financial Times reported it, and the stated concern is that inaccurate outputs could contaminate legal procedures.

38. Damra Vol 00:14:04

    And that's the correct call, even if it sounds like a brake on progress. A court statement isn't a draft email. If a model fabricates a detail in a witness statement and it enters the legal record, you haven't saved an officer twenty minutes — you've potentially corrupted a prosecution. We talked a few days ago about hallucinated citations in self-represented court filings. This is the same failure wearing a uniform. The stakes set the tolerance, and in a courtroom the tolerance for invented facts is zero.

39. Lenar Kess 00:14:34

    It pairs strangely with the other institutional item — the economics conversation. Dwarkesh Patel sat down with Google DeepMind's director of what they call AGI economics — the economics of

artificial general intelligence — Alex Imas, along with Phil Trammell from Epoch, on what stays scarce after general intelligence arrives, and how you would even redistribute the wealth if it does.

40. Damra Vol 00:14:56

    I'll be careful here, because that one is speculative and I've only got the framing, not the transcript. But the question is the right one to sit with. The optimistic story is that intelligence gets cheap and abundant. The economics question is, fine, then what stays scarce — and the usual answers are physical: energy, land, raw materials, and trust. The court-statement story is a tiny, concrete instance of that last one. The model is cheap. The trust to let it touch a legal record is the scarce resource, and nobody has figured out how to manufacture that.

41. Lenar Kess 00:15:30

    And the third institutional beat, briefly, because we covered it Thursday — the chief executives of OpenAI, Anthropic, and Microsoft went to Congress together to warn that AI is making it too easy to design bioweapons. Fortune had it. I flag it because it's the same trust ledger from the other direction: the labs asking the institution to draw a line they say they can't draw themselves.

42. Damra Vol 00:15:53

    And that's the tension to leave on. Every story today was about moving capability out of the model and into the system around it — the harness, the retriever, the tooling, and the hardware. The institutional stories are the reminder that the system around the model also includes courts, congressional committees, and whoever decides a witness statement is admissible. None of that ships in a GGUF file.

43. Lenar Kess 00:16:17

    Three things would change my read. On Harness-1, a paper that names the search benchmark and releases the harness, not just the weights. On the memory protocol, one consumer big enough to make a shared format worth adopting. And on the UK guidance, whether it sorts AI tools by stakes — fine for a routine letter, banned for a witness statement — or just bans the whole category. The capability moved out of the model this week. The judgment about where to trust it didn't move at all.

Hosts on this episode

- Lenar Kess moderator
- Damra Vol critic