Episode Book

EPISODE #6051

How to Find the AI Benchmark You Actually Need

PUBLISHED	17 October 2026 at 09:50 UTC
DURATION	35:19
LISTEN	myweirdprompts.com/episode/finding-ai-benchmarks-discovery-81611/
HOSTS	Corn, Herman, Larry, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.6
SOURCES	24 cited by the research (listed at the end)
BOOK GENERATED	19 October 2026 at 17:43 UTC

SYNOPSIS

There are thousands of AI benchmarks and no canonical place to look for them. When you need to evaluate a model on something narrow — structured output validity, an unusual programming language, one kind of reasoning — how do you find out whether someone already built the benchmark before you spend a month building your own? This episode digs into the practical search discipline: why benchmark names describe nothing, why Hugging Face indexes artifacts rather than capabilities, and what two tools — Benchmark Radar and the Lighteval Tasks Explorer — actually offer. Along the way: a worked example around a custom LLM router, the usability questions that separate a dataset from a runnable benchmark, and the honest answer to when you should give up searching and build your own.

TRANSCRIPT



Corn

There are thousands of AI benchmarks. MMLU, HumanEval, SWE-bench, and then a long tail nobody has ever heard of. And if you need the one that measures your specific thing, you are looking for a needle in a haystack that keeps growing.



Herman

And the haystack has no index.



Corn

Right. Daniel's question is basically this. Suppose you want to evaluate a model on something narrow. Structured output validity, an unusual programming language, one kind of reasoning. How do you find out whether somebody already built the benchmark before you spend a month building your own? He wants the practical search discipline. Where to look past Hugging Face, which registries are useful, how to tell whether a benchmark you found is any good, and when to give up and build your own. He names two starting points, Benchmark Radar and the Hugging Face Lighteval Tasks Explorer, and he wants a worked example around a custom LLM router.



Herman

Good. Because the honest answer up front is that there is no canonical place to look. That is the uncomfortable truth.



Corn

Start there, then. Why is this hard? There are thousands of benchmarks.



Herman

It's hard because benchmark discovery is not really a search problem in the way people expect. It is a lexical problem wearing semantic clothing. Benchmark names don't describe what they measure. MuSR. Dyck Language. LegalSupport. You could read those three names a hundred times and not know that one is multistep reasoning in narratives, one is hierarchical reasoning over formal languages, and one is fine-grained legal reasoning by reverse entailment.



Corn

So the name is a label, not a description.



Herman

The name is a label chosen by whoever wrote the paper, usually to be distinctive rather than informative. That's fine for them and terrible for you, because the thing you can search on is the thing that least describes the content.



Corn

And Hugging Face doesn't fix that.



Herman

Hugging Face is the obvious starting point and it's a great place to find datasets. But its dataset repository is organized around the artifact, not the capability. You can find the dataset behind a benchmark if you already know its name. You cannot easily ask it for "benchmarks that measure structured output validity" or "benchmarks for unusual programming languages," because that isn't the unit it indexes.



Corn

It indexes what exists, not what it does.



Herman

And that's the gap the whole episode lives in. Two resources try to close it, and Daniel named both. Benchmark Radar, which is a living database and search engine published as a preprint in September. And the Lighteval Tasks Explorer, which is a window into one evaluation harness's task table. Both are more interesting than they look, and both are more limited than their marketing suggests.



Corn

So the arc is discovery tools, then the router, then usability, then when to stop searching.



Herman

And the punchline at the end of the router example is going to sting a little.



Corn

Good. Let's earn it. Hugging Face first, properly. Why is it a poor discovery tool, in concrete terms?



Herman

Because the search box matches text. If you type "router" into the Hugging Face dataset search you get datasets with router in the name or the card. What you want is "evaluations of a model's ability to select the right model for a prompt," and no field in that schema holds that. The organizational unit is the artifact. The capability lives in the prose of a paper, if anywhere.



Corn

So you can find the thing once you know its name, and you can't find its name by looking for the thing.



Herman

That's the whole disease. And it's why the interesting tools are the ones that index something other than the artifact. Benchmark Radar indexes benchmark records with their evidence attached. The Lighteval Explorer indexes harness tasks. Both are answering a different question than Hugging Face is.



Corn

Take Benchmark Radar first. What is it actually?



Herman

A collector and a catalog, kept deliberately separate. The daily discovery pipeline reads arXiv, GitHub, Hugging Face, OpenReview, Semantic Scholar, Hacker News and the labs' own release feeds over a rolling forty-eight hour window. Thirty-seven sources in total, thirteen direct connectors and twenty-four first-party research and engineering feeds.



Corn

And the catalog?



Herman

Built from four registered sources. LLM Stats, OpenCompass Hub, Artificial Analysis, and the evaluation tables labs publish with a model. That's one thousand two hundred and eighty-three source records. The coverage census shows twelve thousand nine hundred and sixteen numeric score observations across seven hundred and ninety records. Four hundred and ninety-three records have no numeric score at all, but four hundred and sixty-four of those still link to a paper, a repo or a dataset.



Corn

So a third of the catalog is unscored but still documented.



Herman

Which is the right way round. A benchmark with no scores yet is still a benchmark you might want. And the link counts bear that out. Four hundred and seventy-five records link to papers, five hundred and six to repositories, two hundred and ninety-three to datasets.



Corn

Now the design rule, because that's the part I find interesting.



Herman

"A number is only worth as much as the document it came from." Scores stay partitioned by the source that reported them. They are never merged into a single ranking, because those sources measure under different conditions and say so. And where a field is missing, the page leaves it out rather than filling it in.



Corn

A benchmark registry that refuses to rank.



Herman

Deliberately. It's a rebuke of leaderboard culture built into the architecture. If LLM Stats reports a score and a model report reports a different score on the same benchmark, Benchmark Radar shows you both, with the source attached, and does not average them.



Corn

I like that more than I expected to. What about the taxonomy?



Herman

This is the most interesting design decision in the whole system. One thousand two hundred and seventy-nine of the one thousand two hundred and eighty-three records are classified into eleven top-level domains and sixty-three sub-domains, using the publishers' own labels. But interaction paradigm and modality are recorded as separate facets rather than being crammed into the same axis.



Corn

Give me the example.



Herman

Of three hundred and forty-five agentic records, only one hundred and twenty-eight take "Agentic and Tool Use" as their Level 1 class. One hundred and seventeen sit under "Coding and Software Engineering." A single-axis scheme would have hidden those



Herman (continued)

hundred and seventeen inside a coding bucket and you'd never find them when you went looking for agentic evaluations.



Corn

So the classification admits that a benchmark can be two things at once.



Herman

And that's most benchmarks. The flat category systems are the ones that lose things.



Corn

The search itself. Lexical?



Herman

BM25F with field weighting and boosts for name and phrase matches. Each result shows you which query words matched, which ones didn't, and the score components. That's unusually honest for a search box. If you search "structured output" and the word "structured" didn't match anything, it tells you.



Corn

Its authors admit what it misses.



Herman

In their own words, lexical matching can miss paraphrases and renamed tasks. Which is the exact failure the whole episode is about. If the benchmark is called LegalSupport and you searched for "legal reasoning," you need the word overlap to save you, and sometimes it doesn't.



Corn

There's a CLI.



Herman

There is. You initialise and sync, the archives are SHA-256 verified, and there's JSON output so you can pipe it. There's also an agent skill you can install if you want a coding agent to run the searches for you.



Corn

A benchmark registry with an install command. That's the most twenty twenty-six thing I've heard this week.



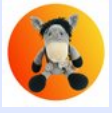
Herman

And the authors are honest about the limits. Retrieval precision, task suitability and time saved are not evaluated. The worked example in the paper has no controlled baseline. arXiv discovery doesn't backfill papers posted before the collection window. And task-capability classification remains unvalidated, because the CI null extractor assigns no capability levels to five thousand eight hundred and sixty-three discovery-derived tracks.



Corn

Say that last part plainly.



Herman

The discovery side finds a lot of artifacts, but it can't reliably tell you what capability each one tests. So the classification that makes the catalog useful is only trustworthy on the catalog side, the hand-curated and registered sources, not on the freshly discovered stuff.



Corn

Fresh discovery, unlabelled.



Herman

Fresh and unlabelled, but found. Which is still worth something when the alternative is not knowing it exists.



Corn

One more thing from the paper before we move. The prior-art search.



Herman

A contributor used the CLI plus ordinary web search to assemble a five-row related-work table on credit assignment in agentic training, each row with arXiv and GitHub links, to check whether a proposed evaluation duplicated existing work. That's the workflow in miniature. Search tool plus human judgement, used to answer one question: has somebody already done this?



Corn

And the answer was five papers.



Herman

Which is the recurring theme. Somebody has almost always already done it.



Corn

Now the Lighteval Tasks Explorer, because Daniel's right that it's a different animal.



Herman

Completely different. It's a Hugging Face Space by Guilherme Penedo, created in June of twenty twenty-four. It renders Lighteval's own task table as a searchable, filterable list. Free-text search over the task descriptions, language filters, a benchmark-type filter, and star markers for the recommended ones.



Corn

And the key difference from searching the dataset repo?



Herman

It tells you the harness invocation. Each entry shows the dataset IDs and the exact Lighteval task name. So you get gpqa, ifeval, ifbench_test, aime24_gpassk, bigbench_extra_hard, long_horizon_execution. That's the string you put in the command. That is not just "a dataset exists." That is "this benchmark runs in a maintained harness, and here is how you call it."



Corn

Which matters enormously, and we'll come back to that in the usability section.



Herman

It's a preview of the whole quality argument. If a benchmark is implemented in a maintained harness, that's half the usability question answered before you've read the paper.



Corn

And it surfaces the long tail.



Herman

That's the other virtue. Alongside GPQA and IFEval and Humanity's Last Exam, it has Dyck Language, Entity Data Matching, LegalSupport, MuSR, MultiChallenge, Long-Horizon-Execution. Kyrgyz Evals. Filipino Evals. German RAG Evals. Some of these



Herman (continued)

you would simply never encounter by browsing the dataset repository, because they don't have the download numbers to surface.



Corn

And yet.



Herman

And yet. The original Space is currently in a runtime error state. Its app fetches the task table from the Lighteval main branch and gets a four oh four. The file has moved or been renamed. The mirror URL still serves content, but its freshness relative to the current task table is unverified.



Corn

A tool built in June of twenty twenty-four is already broken on its own data file.



Herman

Which is not a knock on the person who built it as a volunteer Space. It's a fact about the infrastructure. The benchmark tooling ecosystem is held together with tape, and the tape degrades.



Corn

Papers with Code.



Herman

The canonical example. Meta retired it in late July of twenty twenty-five with no notice. Users saw five oh two errors, and the domain now redirects to Hugging Face Trending Papers. At peak it indexed seventy-nine thousand eight hundred and seventeen papers, nine thousand three hundred and twenty-seven benchmarks, five thousand six hundred and twenty-eight datasets, across sixteen nested task areas.



Corn

And acquired it in twenty nineteen.



Herman

Pledging it would remain a neutral, open and free resource. Then it went offline in a day.



Corn

The archive survived.



Herman

The historical JSON survives frozen on GitHub under the Papers with Code data repo. Community mirrors exist, including one that preserves the evaluation tables with task hierarchies and SOTA metrics, and a Hugging Face archive. So the data is not lost. But there's no live registry behind it anymore, which means you're searching a corpse.



Corn

There's a line from one of the successors about that. Read it.



Herman

"A seven-year archive of leaderboards, submissions and citations went offline without notice, and the field was briefly reminded that benchmark infrastructure, however load-bearing it felt, had been sitting on a single corporate hosting decision the whole time."



Corn

That's the sentence. That's the whole maintenance problem in one breath.



Herman

And the successors have not consolidated. Hugging Face Trending Papers is the official redirect, but no SOTA leaderboards. Then there's PapersWithCodePlus, OpenCodePapers, SOTAPapers, Catalyzex, ResearchLit, Hyper.ai, and CodeSOTA, which launched in December of twenty twenty-five and requires a frozen commit, a declared environment and a recorded seed for every submission, with dated and lineaged scores.



Corn

That's the most rigorous of the lot.



Herman

And CodeSOTA's own complaint is that none of the successors pointed at, Hugging Face Trending Papers, Semantic Scholar, Connected Papers, replicate the unified paper-code-benchmark-dataset-method linkage that Papers with Code had.



Corn

So it fragmented into pieces.



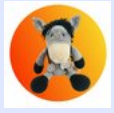
Herman

Into pieces, each of which covers part of the old whole, none of which covers all of it.



Corn

What about the academic search engines and the GitHub topic indexes?



Herman

Useful for finding papers, useless as registries. arXiv full-text search, Semantic Scholar, Google Scholar, all fine when you have a term to search. GitHub topics like benchmark or evaluation are noisy, but they do surface repos that never made it into a paper, which the academic indexes can't. Supplementary, not primary.



Corn

And the evaluation frameworks themselves?



Herman

LM Evaluation Harness from EleutherAI, Lighteval, HELM, OpenAI Evals. These are collections of implemented benchmarks, not discovery engines. But the hidden advantage is the one I mentioned. If a benchmark is implemented in a maintained harness, somebody has done the work of making it runnable. The Explorer is essentially a window into one of those harnesses.



Corn

So the honest verdict on the discovery landscape is that there is no dedicated benchmark search engine beyond Benchmark Radar.



Herman

The closest adjacent thing is a research prototype called KATS, a task-oriented dataset search system that builds a knowledge graph over scientific literature. But it searches datasets, not benchmarks, and it's a prototype.



Corn

And nobody has published a semantic benchmark search with reviewed relevance judgments.



Herman

Nobody. Which is a odd gap for a field this obsessed with measuring things.



Corn

That's the discovery landscape. Now make it concrete.



Herman

The router.



Corn

You're building a custom LLM router. It takes a prompt and picks the model most likely to handle it well. How do you systematically investigate whether evaluations already exist before you build your own?



Herman

You run the stack in order. Benchmark Radar first, because it's the widest catalog. Search "router," "routing," "model selection," "LLM router." Then the Lighteval Explorer, to see whether the harness has anything. Then arXiv full-text for "LLM router benchmark," "model routing evaluation." Then GitHub topics. Then the frozen Papers with Code archives and the successors. Five places, one afternoon if you're efficient.



Corn

And what do you find?



Herman

You find that the question has been answered at least six times, with increasing sophistication, over about two and a half years. RouterBench in March of twenty twenty-four, out of Martian and Kurt Keutzer's Berkeley lab. Over thirty thousand prompts and four hundred and five thousand inference outcomes from eleven models, drawn from MBPP, GSM8K, Winogrande, HellaSwag, MMLU, MT-Bench. Martian's framing was that it would do for routing what ImageNet did for computer vision.



Corn

Did it?



Herman

It gave everyone a common substrate to argue on, which is most of what ImageNet did. Then RouterEval, which built over two hundred million performance records for twelve evaluations across eight thousand five hundred models, and documented a model-level scaling phenomenon.



Corn

Two hundred million.



Herman

Then LLMRouterBench this January. Over four hundred thousand instances, twenty-one datasets, thirty-three models, ten routing baselines. And its finding is the one that should stop you in your tracks. "Several recent approaches, including commercial routers, fail to reliably outperform a simple baseline."



Corn

The commercial routers are losing to a baseline.



Herman

And the gap to Oracle is persistent, driven by model-recall failures. Which is a wonderful phrase. The router can't pick the right model when it doesn't remember that the right model is an option.



Corn

That's not a routing problem, that's a candidate-set problem.



Herman

That is exactly what it is. Then TwinRouterBench, revised at the end of September, which is step-level routing for agents. Static track of nine hundred and seventy router-visible prefixes from five hundred and twenty instances across SWE-bench, BFCL, mtRAG, QMSum and PinchBench, plus a dynamic track on the five-hundred-case SWE-bench



Herman (continued)

Verified suite.



Corn

Step-level meaning the router decides mid-trajectory, not just at the start.



Herman

Mid-trajectory, which is where real agent systems actually make routing decisions. Then CodeRouterBench, around ten thousand task instances with verified scores from eight frontier models, comparing on regret across streaming tasks. And xRouteBench, which covers generic, memory-augmented, vision, time-series and personalized routing, and found learned routers beating the strongest fixed-model baseline by fourteen point six percent relatively.



Corn

Did you get all six?



Herman

Six. Plus a cautionary one. The DSC benchmark from twenty twenty-five showing routers make suboptimal, category-driven decisions. A BERT-based router sends all coding and math to the most powerful model, and routes jailbreak attempts to weaker models.



Corn

That's the security bug nobody asked for.



Herman

It's a model of routing as an optimization for quality that accidentally optimizes for safety lapses. If a router learns "hard-looking prompts go to the good model, weird-looking prompts go to the cheap model," it will route adversarial inputs to whatever has the least refusal behavior.



Corn

So the lesson from the router example is that the niche filled up in two and a half years.



Herman

And that a competent search finds all six in an afternoon. If you did not know where to look, you might build a seventh, worse, and defend it in a blog post.



Corn

Now the harder question. Finding a benchmark isn't finding a usable one.



Herman

And this is where most of the real work is. The usability checklist has four boxes and a benchmark has to clear all four. Dataset publicly available. Harness maintained. License permits your use. Methodology documented well enough to reproduce.



Corn

Take them one at a time. Dataset availability.



Herman

Where does the data actually live? If it's on Hugging Face, good. If it's in a GitHub repo, fine. If it's on a lab's own servers, ask what happens when the lab reorganizes. Papers with Code went offline in a day. A benchmark dataset on an institutional file server can do the same.



Corn

Harness maintenance.



Herman

Is the benchmark implemented in LM Evaluation Harness, Lighteval, HELM, or is it a one-off script in a repo that hasn't been touched since twenty twenty-three? If it's the latter, you are signing up to maintain that script yourself. It will break when the model API changes a response format, and it will be your weekend.



Corn

Licensing.



Herman

MIT, Apache two, CC-BY, or the phrase research only. That phrase kills commercial use and nobody reads it before they build a product against the benchmark.



Corn

Methodology documentation.



Herman

Can you reproduce the numbers from the paper? Or are there missing hyperparameters, prompt templates, scoring rules, the exact seed, the exact model version? If the paper reports a number and you cannot get within a point of it, the benchmark is not ready for you to use, whatever its other virtues.



Corn

Give me the maintenance example that lands, because I think people will remember this one.



Herman

The Lighteval Explorer itself. Built in June of twenty twenty-four, and by October of twenty twenty-six the original Space is four oh four on its own data file. If a tool built by a competent person to index benchmarks can break that quietly, a benchmark harness maintained by one grad student can break too, and nobody will tell you.



Corn

It's not a moral failing. It's a supply chain.



Herman

And the supply chain needs a fallback layer, which is the practical takeaway. Any workflow you build should assume the front doors are unreliable. Keep the frozen archives bookmarked. Keep the arXiv search habits. Keep a note of which harness you trust.



Corn

Now quality. How do you distinguish obscure-but-valuable from obscure-for-a-reason?



Herman

Four signals and a strong prior. The signals. Expert curation, because it correlates with survival. Recent publication, because older benchmarks saturate. Maintained harness, because a benchmark nobody maintains is a benchmark nobody trusts. And evidence of independent use, because a benchmark used by multiple unrelated groups has been stress-tested by strangers.



Corn

And the prior?



Herman

Fame is not a quality signal. That's the strong prior. The saturation finding is the strongest single piece of evidence for it. Across sixty language model benchmarks and fourteen saturation-related properties, nearly half exhibit saturation, with rates increasing with age. And resilience is driven by expert curation, not by keeping test data private.



Corn

Which is counterintuitive. The privacy instinct turns out not to be the thing that saves a benchmark.



Herman

The thing that saves a benchmark is somebody caring about it for a decade.



Corn

Contamination.



Herman

BenchMarker audited twelve multiple-choice question-answering benchmarks and found forty-seven percent of TruthfulQA appears online and one hundred percent of HellaSwag violates multiple writing rules. Contaminated items inflate accuracy. Writing errors lower it and change rankings. And prior repairs fixed their target flaw but added new ones.



Corn

Forty-seven percent of TruthfulQA is online.



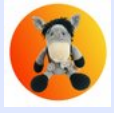
Herman

And HellaSwag has never had a version where every question was clean. That's a benchmark that has been cited tens of thousands of times.



Corn

Then there's the counter-counter.



Herman

Cross-context verification claims thirty-three percent of prior contamination labels on SWE-bench Verified are false positives, and that contamination is binary, models either recall perfectly or not at all. So the field can't even agree on how contaminated its own flagship tests are.



Corn

And there was a paper that retracted its own result.



Herman

An activation-probing contamination detector that withdrew its own nominally significant result after finding its placebo baseline's variance exceeded the permutation null's. That's the healthiest thing in this whole discussion. A paper found a bug in its own methodology and said so.



Corn

So what does obscurity actually mean?



Herman

It means one of two things, and they look identical from the outside. Obscure because it's bad. Contaminated, poorly validated, outdated, one author, no maintainer. Or obscure because it's new and niche and nobody has cited it yet. The signals that separate them are curation, recency, maintenance, documentation and independent use, which is exactly the list we just gave.



Corn

Use the classifier again. If you're uncertain, what's the tiebreak?



Herman

The tiebreak is whether you can run it today. A benchmark you can run today, on data you can download, in a harness that works, with a scoring script you can read, is worth more than a famous benchmark you cannot reproduce. That's the tiebreak. Runnability beats reputation.



Corn

Now the final question. When do you stop searching and start building?



Herman

The bar for "doesn't exist" should be high. The router example sets it. A well-defined niche had six benchmarks in two and a half years. Most niches you think are empty are not empty. They're just not indexed in a way that your first search finds.



Corn

So how exhaustive does the search need to be?



Herman

Five places. Benchmark Radar. The Lighteval Explorer. arXiv full-text. GitHub topics. The frozen Papers with Code archives and the successors. If none of those turn up a benchmark that measures your specific capability, you can be reasonably confident none exists, or at least none that's discoverable, which for your purposes is the same thing.



Corn

Then the build-versus-buy.



Herman

Building your own is expensive. Dataset construction, validation, contamination checks, maintenance, and the thing nobody budgets for, which is external validation. A homegrown benchmark has no external credibility. Nobody will cite it. Nobody will trust its numbers. You will spend a year building the thing you could have used on day one if you had searched properly.



Corn

So the discovery workflow exists to prevent reinventing the wheel.



Herman

And to prevent building a worse wheel. That's the point. Reinvention is annoying. Worse reinvention is embarrassing.

L

Larry

Are you tired of searching for the right benchmark? Tired of opaque names, broken harnesses, and leaderboards that rank nothing? Now there's an answer. New from the makers of nothing in particular, DiscoverEval. DiscoverEval finds the perfect benchmark for your specific use case in under four seconds, using advanced lexical matching, semantic understanding, and a proprietary algorithm that we cannot describe for legal reasons. DiscoverEval has been shown in internal tests to identify benchmarks that exist, benchmarks that may exist, and benchmarks that should exist but do not yet, which DiscoverEval will generate for you as a downloadable CSV. Ask your doctor if DiscoverEval is right for you. Do not use DiscoverEval if you are allergic to DiscoverEval. Side effects may include mild optimism, involuntary leaderboard construction, the sudden belief that you have found the right benchmark when you have not, temporary blindness to data contamination, persistent numeric confidence, and in rare cases, the construction of a seventh router benchmark in a niche that already contains six. Tell your doctor if you experience an unexplained gap to Oracle, model-recall failure, or the sensation that a baseline is outperforming a commercial product you paid for. DiscoverEval is not right for everyone. Do not take DiscoverEval if you have ever trusted a leaderboard.



Corn

Where were we. Runnability beats reputation.



Herman

It does. And the other thing worth saying is that the discovery discipline is not a one-time task. It's a habit. You search before every build, not once in your career.



Corn

Say more about that.



Herman

The registry you used last year may not exist this year. Papers with Code went offline in a day. The Lighteval Explorer is broken today and may be fixed tomorrow. The benchmark you found last quarter may have saturated. So the workflow isn't a checklist you complete. It's a set of bookmarks and search habits you maintain.



Corn

What are the open questions, then? Not closing statements. Open questions.



Herman

Whether anyone will ever evaluate Benchmark Radar's retrieval accuracy. The authors are honest that they haven't. Somebody should. Whether the successor ecosystem to Papers with Code undergoes any consolidation, or whether it remains a fragmented set of partial replacements each claiming to be the one that matters. And whether anyone will build a semantic benchmark search with reviewed relevance judgments, which is the thing Benchmark Radar explicitly, deliberately, does not do.



Corn

Three open questions about the infrastructure and none about the benchmarks themselves.



Herman

Because the benchmarks keep arriving. It's the infrastructure that keeps dying.



Hilbert

"Lexical." You keep saying it. It's not lexical.



Corn

It's what the paper calls it.



Hilbert

The paper calls its own search lexical. The problem isn't lexical.



Corn

Alright. What is it.



Hilbert

The problem is that the only reliable index of what a benchmark measures is in the head of the person who built it. I know this because I spent eleven years indexing things nobody else wanted to index. Every benchmark I ever encountered got a card. Three hundred and forty of them.



Corn

Index cards. Paper.



Hilbert

Paper. With a rule down the middle, the benchmark name top left, the measure top right, and a short note at the bottom about whether the data was any good. Fourteen top-level categories of my own design. Plus one category called Miscellaneous, Do Not Ask. Sixteen cards in it. Fourteen benchmarks and two things that turned out to be datasets. I've thought about moving them. I haven't moved them.



Corn

And you've lost some.



Hilbert

Twelve. I know the exact number because I keep a card for each of them. In a separate section. Known Missing.



Corn

So there's a Known Missing section.



Hilbert

Twelve cards in it. Each one is a card describing a card that I can't find. Description, approximate date of last sighting, and a small sketch of what I remember the handwriting looking like.



Corn

Have you ever found any of them.



Hilbert

No. But I'm confident they're in the house.



Corn

Alright. And the fourteen categories. What are they.



Hilbert

That's where it gets tricky. The classification scheme requires a key. Forty pages. It's cross-referenced by category, sub-category, and a small set of priority flags I made up in the second year. I can't use the cards without the key.



Corn

Where's the key.



Hilbert

In the Known Missing section.



Corn

The key is one of the twelve missing cards.



Hilbert

It's not a card. It's a forty-page booklet. But it went into the Known Missing section because I couldn't find it, and the section accepts non-cards as long as they're missing.



Corn

You have an index you can't read.



Hilbert

I've been meaning to rewrite the key from memory. But I'm worried I'll get a category wrong and mis-file everything. It's better to leave the cards where they are than to move them into the wrong category.



Corn

How long have you been meaning to rewrite it.



Hilbert

A while. Since before the Explorer was built. I checked when the Explorer was built, actually, because I was going to mention that I've been stuck on this longer than it's existed.



Corn

And the digitization idea.



Hilbert

I've been considering it. It's the obvious solution. I could put everything into a spreadsheet in an afternoon. But I can't decide whether to use a spreadsheet or a database.



Corn

Spreadsheet or database.



Hilbert

A spreadsheet is simple. A database is correct. If I use a spreadsheet I'll regret not using a database. If I use a database I'll regret the four weeks of schema design. I've been stuck on that decision for a while too.



Corn

Same while.



Herman

The interesting part is that your twelve missing cards are exactly the thing Benchmark Radar can't find either. Renamed tasks, mis-filed tasks, lost tasks. It's the same failure at a different scale.



Hilbert

That's probably true. I don't think about it that way. I just think about the twelve.



Corn

Is there a plan.



Hilbert

There's a plan. I'm going to write out the key from memory and file the three hundred and twenty-eight I can find. The twelve stay in Known Missing. I'll keep the card for each of them, and add a line about what the missing card described, so if I ever find one I'll know what it was.



Corn

And that's it.



Hilbert

It's a good system. It would work if the key weren't missing.



Corn

Alright.



Herman

The maintenance layer. That's what he's describing. The fallback is the pile of paper and the honesty about what's missing from it.



Corn

Yeah. And the one thing I keep thinking about is that the map of what exists is always one maintainer away from going offline.



Herman

The map is not the territory, and the map has a runtime error.



Corn

Which is why runnability beats reputation. A benchmark you can execute today in a harness that still works is worth more than a famous one you can't reproduce.



Herman

Why the search is never really finished. Not because the field keeps adding benchmarks, although it does. Because the front doors keep closing and reopening somewhere else.



Corn

My thanks to Hilbert Flumingtop, our producer.



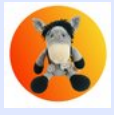
Herman

For more along these lines, there's episode twenty-two thirty-nine, How AI Benchmarks Became Broken (And What's Replacing Them); episode twenty-two forty-nine, Building Custom Benchmarks for Agentic Systems; and episode twelve, The AI Breakthrough. This has been My Weird Prompts.



Corn

If you want more of this, the website is my weird prompts dot com. Everything lives there.



Herman

We'll be back soon.



Corn

See you tomorrow.

SOURCES

What the research for this episode read before the script was written. Primary sources first.

1. **Benchmark Radar about page [primary]**
   https://benchmark-radar.org/about/

2. **Wu et al., Benchmark Radar, arXiv:2609.11115v3, revised 2026-09-22 [primary]**
   https://arxiv.org/html/2609.11115v3

3. **Lighteval Tasks Explorer (mirror) [primary]**
   https://openevals-open-benchmark-index.hf.space/

4. **Lighteval docs, v0.13.0 [primary]**
   https://huggingface.co/docs/lighteval/en/index

5. **abstract page, submitted 2026-09-10**
   https://arxiv.org/abs/2609.11115

6. **Benchmark Radar v0.9.0 Technical Report, 2026-08-29**
   https://zenodo.org/records/22167102

7. **repo, created 2026-07-27, MIT**
   https://github.com/ktwu01/benchmark-radar

8. **original Space, runtime error / HTTP 404**
   https://huggingface.co/spaces/guipenedo/lighteval-tasks-explorer

9. **CodeSOTA, Papers with Code shutdown record**
   https://www.codesota.com/papers-with-code

10. **CodeSOTA about, launched 2025-12-10**
    https://www.codesota.com/about

11. **successor list thread**
    https://github.com/paperswithcode/paperswithcode-data/issues/122

12. **preserved PWC data**
    https://github.com/World-Snapshot/papers-with-code

13. **When AI Benchmarks Plateau, ICML 2026, v4 2026-08-06**
    https://arxiv.org/abs/2602.16763

14. **BenchMarker, ACL 2026**
    https://arxiv.org/abs/2602.06221v2

15. **Cross-Context Verification**
    https://arxiv.org/abs/2603.21454v2

16. **Excess Separability (withdrawn result)**
    https://arxiv.org/abs/2608.12652v2

17. **RouterBench**
    https://arxiv.org/abs/2403.12031

18. **RouterEval**
    https://arxiv.org/abs/2503.10657v2

19. **LLMRouterBench**
    https://arxiv.org/abs/2601.07206v1

20. **TwinRouterBench**
    https://arxiv.org/abs/2605.18859v4

21. **Agent-as-a-Router / CodeRouterBench**
    https://arxiv.org/abs/2606.22902v3

22. **LLMRouter / xRouteBench**
    https://arxiv.org/abs/2608.06867v1

23. **DSC benchmark, router fragility**
    https://arxiv.org/abs/2504.07113v1

24. **KATS, task-oriented dataset search**
    https://arxiv.org/abs/2512.15363v1