Episode Book

EPISODE #5857

Building a Scorecard for Our Own AI Writers

PUBLISHED	11 October 2026 at 10:43 UTC
DURATION	32:57
LISTEN	myweirdprompts.com/episode/bespoke-llm-benchmark-script-agent-77594/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.3
SOURCES	17 cited by the research (listed at the end)
BOOK GENERATED	19 October 2026 at 17:40 UTC

SYNOPSIS

Every episode of this show is written by an agentic pipeline, and for most of the archive that script-writing agent has been DeepSeek V4, with Gemini, Claude, and a stretch of OpenRouter randomization in the mix. So when Daniel asked how to build a bespoke benchmark for it, the answer couldn't be vibes and one careful listen. This episode works through the three things he wants measured — mid-episode repetition, lifelike dialogue between two named recurring hosts, and whether the retrieved sources actually make it into the script — and why no existing benchmark covers that combination. EQ-Bench's creative-writing suite explicitly says it can't assess conversational roleplay and is English-only. HELM measures SQuAD accuracy. The closest neighbours are rubric-plus-Elo designs and composable binary-leaf rubrics, plus two enterprise agentic-grounding harnesses. We dig into rubric saturation, judge self-preference bias, position bias, and the case for deterministic n-gram scoring where no judge model is needed at all.

TRANSCRIPT



Corn

Quick thing before we start. Every episode of this show is generated by an agentic pipeline. A planning agent picks up Daniel's prompt, delegates to research sub-agents, pulls from a lorebook and a retrieval store over past episodes, and then hands the whole thing to a script-writing agent, which is the agent that actually matters — the one that writes Herman and me.



Herman

That last one has been DeepSeek V4 for the overwhelming majority of the archive, with Gemini and Claude in the mix, and a stretch of OpenRouter experiments where Daniel randomized prompts across ten models just to hear what came out.



Corn

And that randomization is where the prompt starts. Daniel's asking how to build a bespoke benchmark for that script-writing agent. Not vibes, not one careful listen. A real scorecard. Three things he wants measured: whether the script degrades into repetition in the middle of the episode, whether the dialogue between Herman and me actually sounds lifelike and fun or just robotic, and whether the agentic harness is doing anything — did the retrieved sources make it into the script, or did the model quietly fall back on parametric knowledge and ignore the inputs.



Herman

And then the second half, which is where it gets interesting.



Corn

Then he wants to take that scorecard, feed it alongside API cost data to a model, and get a cost-benefit analysis out the other end that names the model performing best for the most reasonable cost. And he wants it cheap enough to re-run every time a promising new model ships. His words: "so that the next time we want to do this, it's not going on vibes and my impressions of a single episode."



Herman

He also said something I want to flag early, because it sets the whole tone. He's tempted by the idea that a much more expensive model, one of the latest Anthropic releases, would write a superior script. And his experience argues against it. The Claude scripts came out overly complicated.



Corn

The reason this is harder than it sounds is that the benchmark he wants doesn't exist. The closest thing to it in the wild is creative-writing evaluation, which is a different animal.



Herman

Different animal is right. The llm-eval-suite repo says it flatly: standard LLM benchmarks are built for research papers, not production. HELM measures accuracy on SQuAD and TruthfulQA — and that tells you nothing about how a model performs on your specific task. That's the whole argument for going bespoke.



Corn

So the general leaderboards are measuring the wrong thing. What about the purpose-built creative-writing ones?



Herman

They get closer and they still miss. The EQ-Bench Creative Writing README concedes two things about itself in the first screen: it doesn't assess conversational roleplay skills, and it's English-only. Herman-and-Corn dialogue quality is exactly the thing it says it can't see. And that's the best-in-class public creative-writing benchmark.



Corn

So to be clear about what we found looking for a fit. There are creative-writing benchmarks — EQ-Bench, WritingBench, LitBench, Judgemark, FictionEval. There are agentic-grounding benchmarks — RegLLM, FORCE-Bench. Nothing evaluates a two-host conversational podcast script with named recurring characters plus agentic-harness grounding baked into the same rubric.



Herman

Nothing. So the benchmark has to be built from the failure modes Daniel already knows in his bones, borrowing the architecture from the closest neighbours. EQ-Bench's rubric-plus-Elo setup, and the binary-rubric design out of the Creative-Writing-Rubrics repo.



Corn

Which is a nicer way of saying the scorecard is the episode. So if we're building this from scratch, what actually goes on it — and how do you define each criterion so it's repeatable instead of impressionistic?



Herman

The trap you hit first is rubric saturation. You build a one-to-ten scale, you run DeepSeek, Claude, and Gemini through it, and they all score eight or nine. The rubric has failed to discriminate. EQ-Bench names it outright: rubric scores "can saturate at high performance levels." That's why they bolted pairwise Elo onto the side — head-to-head comparison is "more discriminative, especially at the top end." When all your candidates are strong, the scale has to be able to tell strong from strong.



Corn

So the scale design is the first real decision, and it comes before any criterion.



Herman

Right, and the fix I'd actually argue for is the binary-leaf design. The Creative-Writing-Rubrics package — HBQ-RS — does it differently from a one-to-ten rubric. It's composable binary-question rubrics. The judge answers one yes/no leaf at a time, and aggregation is code, not another model call. It ships two hundred and seventy-seven modules, two thousand one hundred and thirty-nine atomic leaves, eighty-five bundle presets, deterministic scoring, and four possible verdicts per leaf: yes, no, not applicable, cannot assess.



Corn

Say more about why binary leaves beat a numeric scale for this.



Herman

Because "rate this dialogue eight out of ten" is asking a model for a vibe with a number attached. "Does Herman's line in this exchange contain information Corn didn't already have?" is a yes or a no, and a hundred of those aggregated in code give you a repeatable score. It removes the model from the scoring step. The one thing the judge does is answer leaves. Math does the rest.



Corn

And the cautions in that repo map onto exactly the ways it goes wrong we'd hit. Do not reward length, verbosity, ornament, or bland compliance by default. Do not mix user taste into craft scores.



Herman

The second one is the one Daniel will fight. He's a human listening to scripts. His taste is the whole reason he's doing this.



Corn

Well — that's the point of the caution. Taste is a separate field. A column in the table you fill in yourself. But it doesn't go inside the craft score, because the moment it does, the score stops being comparable between runs.



Herman

Agreed, and that's the design principle behind all of it. Score the thing objectively, then record your taste as a distinct column. Two artefacts, not one.



Corn

All right, criterion one. Context degradation and repetition. This is the one Daniel already knows is his weak spot. His phrasing: "repetition in the middle of the episode where models are most prone to losing the thread."



Herman

And it's the criterion I'm most optimistic about, because it's the only one you can score with zero judge models in the loop. The relevant literature is old and settled. Holtzman et al., "The Curious Case of Neural Text Degeneration," showed decoding strategy alone "can dramatically effect the quality of machine text." Repetition In Repetition Out from NeurIPS 2023 frames neural text degeneration as "generating repetitive and dull loops." Nobody is arguing about whether the phenomenon exists. So you measure it deterministically.



Corn

With what.



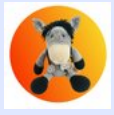
Herman

N-gram repetition rate. Distinct-n — how many distinct unigrams, bigrams, trigrams appear across the script. And self-BLEU between mid-episode segments — take the script, cut it into chunks, and measure how similar the chunks are to each other. If the second half of the episode is regurgitating the first half, self-BLEU spikes and you see it in a number. No Claude, no GPT, no judge model, no inference cost. Just a script and a few lines of code.



Corn

That's the cheapest criterion on the card and it's also the one targeting the breaking point Daniel actually sees. That's a good place to start.



Herman

And it's not a proxy for the real thing. It is the real thing. The thing he's listening for in a mid-episode drift is literally n-gram repetition and declining lexical variety. The metric measures it directly.



Corn

Fine. Criterion two — and this is the hard one. Lifelike and fun dialogue between Herman and me.



Herman

There is no validated automatic metric for this. I looked. The literature offers rubric-based craft criteria. Judgemark has dimensions called "Nuanced Characters" and "Emotionally Engaging." HBQ-RS has binary leaves like "does this line contain a specific detail only this character would know." That's the closest you get to a published operationalization of "does the dialogue feel alive." Everything else is human-scored or hand-waved.



Corn

Which means this criterion needs either a human or a very carefully calibrated judge model.



Herman

Correct, and the calibration is where the landmine is. Self-Preference Bias in Rubric-Based Evaluation found that judges can be more than fifty percent more likely to incorrectly mark a rubric as satisfied when the output is their own. On subjective rubrics, self-preference bias skews model scores by up to ten points. The paper calls that "a potentially decisive margin when ranking frontier models."



Corn

Ten points. That's the entire gap between the best model and the fourth-best model.



Herman

And it means the naive setup — Claude Sonnet judges the Claude script, GPT judges the GPT script — produces a leaderboard that's measuring which model flatters itself hardest. The fix is mechanical: the judge must not share a model family with the generator. If Claude wrote the script, Claude doesn't grade it. Cross-family judges. Or an ensemble.



Corn

And an ensemble doesn't fully fix it either.



Herman

It does not. Ensembling helps and does not eliminate. So the score you get from an LLM judge on this criterion is a soft number, not a hard one. Which is fine, as long as you treat it that way. The repetition criterion is a hard number. The dialogue-quality criterion is a soft number. They go in the same table, and you don't pretend they have the same precision.



Corn

There's another bias in here — position bias. Am I More Pointwise or Pairwise? found that rubric-based judging implicitly resembles a multiple-choice setting and therefore exhibits position bias. Some judges favour the first option. Some favour the last. And the ordering of the criteria themselves shifts scores.



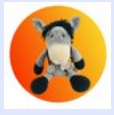
Herman

Which is why you run comparisons in both orders — A before B and B before A — and average. That's one of the EQ-Bench controls. Truncate outputs to four thousand characters to control length bias. Run comparisons both directions to control position bias. Add criteria that penalize verbosity and poetic incoherence, because a model that writes something that sounds profound but parses as nonsense is a real pitfall and a lot of judges give it points.



Corn

What does EQ-Bench explicitly say it does not control for?



Herman

Judge self-bias, positivity and negativity bias, and "slop" bias. Three of the four are on its own list of things it knows it doesn't handle. That's honest of them, but it's also the list of hazards Daniel has to close himself.



Corn

So criterion two is: build the rubric, cross-family the judge, control for position and length, average over both orders, and validate the entire business against Daniel's own ear before trusting an LLM's score on any of it.



Herman

Validate against a small human sample first. That's the last non-negotiable step. If the judge's score and Daniel's score don't correlate on fifty episodes, the judge is measuring something else.



Corn

Criterion three. Is the agentic harness actually working?



Herman

This is where Daniel's question is sharpest, and where we have the least support from the literature. His question: did the retrieved sources make it into the episode, or did the model ignore the inputs and fall back on parametric knowledge? There is no published benchmark that measures that in a creative-generation context. There are two that come close, and both come from agentic AI in enterprises.



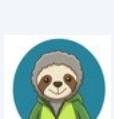
Corn

RegLLM and FORCE-Bench.



Herman

RegLLM is a diagnostic harness for regulated agentic AI, and it instruments exactly the signals Daniel cares about: citation validity, source grounding, schema compliance, escalation correctness, constitutional alignment, unsafe-action rate. The key architectural insight is that it separates programmatic verifiers from AI-judge scores. Some of those checks are code. Some are a model asked to judge. The two are reported separately, because they have different reliability.



Corn

So citation validity is a code check.



Herman

It can be. Did the script cite a source that was in the retrieved set — yes or no. Did it cite a source that was never retrieved — that's a hallucinated citation and it's a deterministic flag. Did the retrieved source's specific claim appear in the script's content — that's a matching problem, harder, but still mostly mechanical for named entities and quoted phrases.



Corn

And FORCE-Bench?



Herman

FORCE-Bench evaluates agentic systems across eight rubric dimensions — accuracy, citations, clarity, depth, groundedness, recency, relevance, structure. "Groundedness" is the dimension that lines up with Daniel's question. It also has a "recency" dimension, which matters if you're asking whether the pipeline is using fresh retrieval or stale training data.



Corn

This is the criterion that most cleanly separates "the model is smart" from "the model is doing the job we gave it." Those are different things, and the pipeline only cares about the second.



Herman

And there's a paper that makes the case for scoring the whole harness rather than just the model. Pufibara Modelica was a harness comparison. Same DeepSeek v4 Flash backend in both. Different harness. With the Pufibara harness, two hundred and two tasks passed. With Claude Code as the harness, one hundred and eighty-five. Same model underneath, seventeen tasks' difference from the scaffolding.



Corn

And that's what Daniel's round-robin OpenRouter function was missing, structurally. It changed the model and kept everything else fixed, which controls for the model — but it doesn't tell you whether the harness would have failed the same model differently.



Herman

Or whether one model is better at using the harness than another is. That's a real interaction. A model that's good at following tool-use instructions will make the retrieval layer look better than it is. A model that ignores context will make the harness look broken when the harness is fine. You have to score the pipeline end-to-end for this criterion, or you can't attribute the score.



Corn

So criterion three is: score the harness end-to-end. Did retrieved sources show up in the episode. Did the model invent citations to sources it never saw. Did it use fresh retrieval or fall back on its training. Score those with code where you can, and only reach for a judge on the residue.



Herman

And one caution from HBQ-RS that applies everywhere on the card. Do not treat CANNOT_ASSESS as NO. If a criterion can't be evaluated for a given script — the retrieval log is missing, the source was ambiguous — you score it as not assessed. You don't round it down to failure. Otherwise your benchmark punishes the pipeline for your own missing data.



Corn

All right, so that's three criteria and a scoring architecture. Card's drafted. But we haven't talked about whether to trust any of it yet. What did LongJudgeBench find?



Herman

LongJudgeBench is the paper that should make everyone nervous. EMNLP 2026 main, one thousand nine hundred and forty-four instances, average output length over nine thousand tokens — long-form judging, which is exactly what we're doing. Across thirty-two model-setting combinations, only twelve exceeded zero point six zero accuracy against human judgement. The overall mean was zero point five six three nine. Random pairwise baseline is zero point five. So the average judge configuration is about six points above a coin flip.



Corn

And creative writing was one of the hard scenarios.



Herman

Creative writing — WP-Bench — scored zero point five seven seven six. Barely better than random. Which is the criterion Daniel cares about most.



Corn

So the paper is telling us an LLM judge on the "is the dialogue fun" question is only slightly more reliable than flipping a coin and calling it.



Herman

On the average configuration, yes. And there's an important nuance: the paper found that references and rubrics help, but not universally. Reference-based judging was best overall, ranging from zero point five three three four to zero point five eight six six. But combining reference plus rubric came out at zero point five seven eight six — slightly worse than reference alone.



Corn

That's counterintuitive. More guidance made it worse.



Herman

Marginally, yes. And the paper documents why: judges get misled by superficial coverage. There's a specific example where a twelve-thousand-token answer scored nine point two six by an LLM judge against a human score of five point eight seven. The judge rewarded length and comprehensiveness of surface. The humans caught that the answer was shallow. So a rubric that includes a "does it cover the topic" leaf can be gamed by padding.



Corn

Which is the exact snag HBQ-RS's first design caution is trying to prevent. Do not reward length, verbosity, ornament, or bland compliance.



Herman

Same disease from two different papers.



Corn

So what's the practical takeaway from LongJudgeBench? Are we saying don't use an LLM judge?



Herman

We're saying don't use an LLM judge as a source of truth. Use it as a fast, noisy signal that has to be calibrated against a human baseline. Use a strong judge model. Provide the anchors — either a reference good script or a well-constructed rubric, and don't assume that stacking both helps. And validate the judge against Daniel's own scores on a calibration set before you trust the judge on anything.



Corn

Okay. So we've got criteria that don't saturate and a judge we can maybe trust. The next problem is running the thing — how many prompts, how many models, how do you turn scores plus costs into a decision. What does EQ-Bench actually run?



Herman

Thirty-two distinct writing prompts, three iterations each, so ninety-six items per model. Generation at temperature zero point seven and min_p zero point one, which EQ-Bench says is "to encourage creativity while maintaining some consistency." Cost is roughly ten dollars per model using Claude Sonnet four point six as judge. Outputs truncated to four thousand characters for length-bias control.



Corn

Three iterations per prompt — that's the direct answer to Daniel's variation worry. His line was about "a single episode" being what he was basing impressions on. Three iterations gives you a within-prompt variance, so you know whether a bad score is a bad model or a bad draw.



Herman

And the prompt count is the thing you scale down. For a podcast, you don't need thirty-two briefs. You need eight to twelve that are representative of the actual distribution of episodes the show produces — one heavy technical, one light, one historical, one with a lot of retrieval, one where the prompt is thin. Eight to twelve briefs times three iterations times N models. That's the run.



Corn

And the scoring step is not a single number. What's EQ-Bench's architecture?



Herman

Two-stage. Stage one is rubric scoring in isolation — each script gets scored on its own, no comparison. Stage two is sparse pairwise — take each model's scripts, run head-to-head matchups against a handful of neighbouring models on the leaderboard, and compute a Glicko-2 Elo weighted by the win margin. Iterate until positions stabilize.



Corn

Why bother with both.



Herman

Because they fail in opposite directions. Rubric scoring in isolation is stable but saturates at the top. Pairwise comparison doesn't saturate but is noisy and has the position bias we just talked about. Together: rubric gives you a within-model score, pairwise gives you a between-model ranking. You're using each to cover the other's blind spot.



Corn

And this is exactly what Daniel's situation calls for, because DeepSeek, Claude, and Gemini are all strong. A one-to-ten rubric is going to put them all at eight or nine and tell him nothing. A pairwise matchup — pick two scripts from two models, ask which sounded more alive — that comparison can tell him something a saturated scale can't.



Herman

You run it both directions. A before B, then B before A, and average. That's the position-bias control and it comes for free once you're doing pairwise.



Corn

Right. Now the cost side, which is where I think the reframing actually lands. Daniel framed his use of DeepSeek as "partially yes for economic reasons." But that's not how the economics work.



Herman

The paper to reach for is Khosravi and Huo, "Which LLM for Which Work? Budgeted Model Allocation under Uncertain Evaluation." Their framing is: given a table of model quality on your task, allocating models to jobs is a multiple-choice knapsack problem. And the hard part is not solving the knapsack. It's estimating the table. They name two ways it goes wrong: models are rarely compared on the same work, and the recorded score is usually a proxy rather than the outcome the company values.



Corn

That second one is the one that argues for this whole benchmark existing. The proxy Daniel has right now is his ear, on one episode, once. The outcome he actually values is how many episodes came out good. Those are different measurements.



Herman

The practical framework is Loaded Cost Per Result — LCPR. Formula is: inference cost plus evaluation cost plus human cost plus operational cost plus a delta term, divided by accepted-work units. The point of the loaded part is that it counts everything, not just tokens. Their worked example is a support-ticket pipeline. A thousand tickets in, eight hundred and twenty accepted. Loaded cost one hundred forty dollars sixty-five. LCPR seventeen cents per accepted ticket, versus a naive fourteen cents per raw ticket. And the anatomy is the punchline: human escalation is seventy-one percent of the loaded cost.



Herman (continued)

Inference is ten percent.



Corn

The naive metric and the loaded metric differ by a factor of twelve, and the biggest cost bucket isn't the model at all.



Herman

It's the rework. And this is the finding that should land for Daniel. LCPR's headline: "On quality-sensitive workloads, a ten-point drop in eval pass rate moves LCPR more than a two times change in per-token pricing."



Corn

Say that again slowly, because it's the whole episode in one sentence.



Herman

On quality-sensitive workloads, the pass rate dominates. Ten percent more episodes surviving, and the total loaded cost drops. Halve the token price and the total loaded cost barely moves, because tokens were ten percent of the loaded number to begin with.



Corn

Daniel's round-robin function, run through this lens, is optimizing the wrong variable. It's been doing knapsack solving on the token-price axis while the actual cost is spread across regeneration, cleanup, and his own time listening and deciding whether an episode is good enough.



Herman

Which reframes the benchmark. The benchmark's payoff is not "which model has the cheapest per-token rate." It's "which model minimizes rejected and regenerated episodes." DeepSeek being cheap is a rounding error. DeepSeek not drifting into repetition in the second half and forcing Daniel to regenerate is the number that matters.



Corn

That aligns with the Khosravi point about information: on paid software tasks, better information about model quality yields more savings than further optimization of the assignment on the same estimates. The scorecard is the high-leverage artefact. Feeding a solver garbage estimates gives you an optimal allocation of garbage.



Herman

Which is the answer to Daniel's second question — how do you combine the scorecard with API cost data and hand it to a model. The AI's job is to solve the knapsack given a good quality table. It is not to guess the quality table. If the estimates are vibes, the solver's output is vibes with decimal places.



Corn

What's the actual shape of the run Daniel should build. Concretely.



Herman

Eight to twelve representative briefs. Three iterations each. Run every candidate model through the same generation call, temperature zero point seven, min_p zero point one. Score every script on three axes — repetition is deterministic code, dialogue quality is a cross-family judge with a human-calibrated rubric, harness grounding is mostly code with a judge on the residue. Then run pairwise Elo among the models using the same scripts. Then take the quality table, multiply by the loaded-cost formula, and let a model solve the knapsack. The output is a ranking plus a per-model loaded cost per accepted episode. That's the deliverable.



Corn

Maintenance. The benchmark has to be cheap enough to re-run casually, because the whole premise is that a new promising model shows up every few weeks.



Herman

EQ-Bench's number is the target: ten dollars per model with a Sonnet four point six judge. The binary-rubric design makes it cheaper, because aggregation is code, not another model call. And the deterministic repetition criterion is free — it's a Python script. So the cost per new model is dominated by the judge model on the dialogue criterion, and that's been getting cheaper.



Corn

The harness criterion might be the cheapest of the lot, because Daniel already has the retrieval logs. The agentic pipeline on Modal is emitting exactly the artefacts you'd need. Which sources were retrieved, which were cited, which made it into the script. That's an offline check.



Herman

Yeah. And the round-robin OpenRouter function he already built is the right instinct — it just needed the scorecard bolted onto it. The pipeline's already doing the experiment. It just wasn't recording the result.



Corn

Let's talk about the price data, because it's concrete and it shapes the decision. DeepSeek V4.1-Flash, off-peak: cache-hit input at three tenths of a cent per million tokens. Cache-miss input at fifteen cents per million. Output at sixty cents per million. Peak is exactly double. Peak windows are Monday to Friday, one to four in the morning UTC and six to ten in the morning UTC. That's seven hours per weekday. About twenty-one percent of the week. Weekends are entirely off-peak.



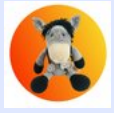
Herman

The cuts against the previous generation are worth the walk. V4.1-Flash versus V4-Flash: cache-hit input down fifty-seven percent, cache-miss input down thirty-two percent, output down nine percent. V4.1-Flash versus V4-Pro at peak: cache-miss input down seventy-seven percent, output down seventy percent, cache-hit input down eighty-six percent.



Corn

The price differential between the mid-tier and the top-tier model is not a factor of two. It's closer to five to one on the output side at peak.



Herman

Here's where the whole comparison runs aground on the earlier reframing. If the loaded cost of an accepted episode is dominated by rework, then the model selection has to be driven by the pass rate, not the per-token rate. A model that costs five times more per token and passes eighty percent of episodes versus one that costs one-fifth and passes sixty percent — the loaded-cost math can go either way, and you can't know without the



Herman (continued)

scorecard.



Corn

The LongJudgeBench finding lines up here. Stronger general-purpose capabilities may not always translate into stronger long-form performance. Daniel's Claude-scripts-were-overcomplicated complaint isn't just him being cheap. The literature supports the idea that a more capable general model doesn't automatically write a better conversational podcast script.



Herman

The Pufibara result is the same shape. Two hundred two versus one hundred eighty-five, same model, different harness. Capability at the model layer doesn't predict outcome at the pipeline layer.



Corn

What's the actual recommendation for the run structure, given Daniel's budget and cadence?



Herman

Run ten briefs. Three iterations. Five to seven models at a time — anything in Daniel's OpenRouter shortlist plus DeepSeek. Score everything. Run pairwise. Do a human calibration pass on twenty scripts to check the judge. Then feed the table into the knapsack solver with the price data, and let it output a ranked list of models by loaded cost per accepted episode. Re-run when a new model ships that looks plausibly competitive. Budget: well under a hundred dollars per run if the judge stays in the mid-tier.



Corn

The maintainability part is what makes it useful. The whole failure with the round-robin was that run after run produced data and no decision, because there was no table to write the data into. The scorecard is the table.



Hilbert

The scorecard's the right idea.



Herman

Sorry, Hilbert, what?



Hilbert

I said the scorecard's the right idea. You left something off it.



Corn

Go on.



Hilbert

I kept a notebook of episodes that sounded off. Not the information — the sound of it. I could tell you which model wrote an episode by the third exchange between the two of you. I had a colour-coded system. The colour for the model that repeated itself was a very specific shade of beige.



Herman

What was the criterion?



Hilbert

Whether it sounded like it was written by someone who was having a good time. That's the one you didn't put on the card. I scored every episode on that alone, one to five. My scores never matched the model labels. The expensive models did fine. A couple of the cheap ones did better. The one that repeated itself was not the one you'd expect.



Corn

You still have the notebook?



Hilbert

I do. I mailed a copy of it to a model provider's customer support address once. Cover letter explaining their beige problem. Never heard back.



Herman

The notebook's just in a drawer somewhere?



Hilbert

It's in a fireproof safe. My neighbour's dog ate the first three pages, so I moved it. I've started a second one, for the other podcast I'm producing. Same show, from the perspective of the microphone.



Corn

The perspective of the microphone.



Hilbert

It's mostly listening. Anyway. Score whether it sounded fun. I have the data.



Corn

Well. The one thing that keeps coming back from all of this is that no benchmark is perfect, and the EQ-Bench README says it better than we could. "Always view benchmark scores as a guide, not absolute truth. Read the sample outputs." And: "no benchmark is perfect. Always supplement scores by reading sample outputs and forming your own judgment."



Herman

Which is the whole design brief for Daniel's scorecard. The scorecard is not the decision. The scorecard is the evidence. The decision is still his, made with a table in front of him rather than a memory.



Corn

There's still a open question about how the dialogue-quality criterion gets stable. Nobody has published a validated automatic metric for "does this sound like two characters who like each other." The lorebook mechanism has no matching benchmark anywhere we can find. So that part is bespoke by necessity.



Herman

As new models drop, the benchmark has to stay cheap enough that running it is the default and skipping it is the exception. Ten briefs, three iterations, a mid-tier judge, and an offline repetition check. That's a system that can live alongside the pipeline rather than fight it for attention.



Corn

Thanks to our producer Hilbert Flumingtop. If you want more of this, try episode nine, Benchmarking Custom ASR Tools - Beyond The WER; episode seven, Building Custom ASR Tools; and episode twenty-two forty-nine, Building Custom Benchmarks for Agentic Systems. This has been My Weird Prompts. If you want to send us a prompt, use the Telegram bot — t dot me slash MWP listener bot. And if you enjoyed this one, a review wherever you found us helps more than you'd think.



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. **EQ-Bench Creative Writing Benchmark v3 README [primary]**
   (2025; rubric + Glicko-2 Elo, 32 prompts × 3 iterations)
   https://github.com/EQ-bench/creative-writing-bench/blob/main/README.md

2. **HaileyStorm/Creative-Writing-Rubrics (HBQ-RS) [primary]**
   (created 2026-08-20; binary-question rubrics, 2,139 atomic leaves)
   https://github.com/HaileyStorm/Creative-Writing-Rubrics

3. **The LCPR Calculator (Sohail Mohammad) [primary]**
   (2026-04-29, updated 2026-07-21)
   https://sohailmo.ai/lcpr-calculator-v2/

4. **jrajath94/llm-eval-suite [primary]**
   (custom-rubric LLM-as-judge with per-criterion breakdowns)
   https://github.com/jrajath94/llm-eval-suite

5. **DeepSeek-V4 Preview announcement [primary]**
   (2026-04-24)
   https://www.deepseek.com/en/news/v4-preview/

6. **Benchmarking LLM-as-a-Judge for Long-Form Output Evaluation (LongJudgeBench)**
   (submitted 2026-06-01, rev. 2026-08-28; EMNLP 2026 main)
   https://arxiv.org/abs/2606.01629v4

7. **Am I More Pointwise or Pairwise? Revealing Position Bias in Rubric-Based LLM-as-a-Judge**
   (2026-02-02, rev. 2026-09-09; EMNLP 2026 Findings)
   https://arxiv.org/abs/2602.02219v3

8. **Self-Preference Bias in Rubric-Based Evaluation of Large Language Models**
   (2026-04-08, rev. 2026-08-03)
   https://arxiv.org/abs/2604.06996v3

9. **Which LLM for Which Work? Budgeted Model Allocation under Uncertain Evaluation**
   (2026-08-30)
   https://arxiv.org/abs/2608.29560

10. **DeepSeek-V4.1-Flash Pricing Explained (apidog)**
    (2026-09-10)
    https://apidog.com/blog/deepseek-v4-1-flash-pricing/

11. **DeepSeek API Pricing (April 2026)**
    (verified 2026-04-24)
    https://deepseekai.guide/api/deepseek-api-pricing/

12. **DeepSeek Raises API Prices And Splits Billing Into Peak And Off-Peak Windows (blinkedtwice.ai)**
    (2026-08-14)
    https://www.blinkedtwice.ai/newsletter/2026-08-14-deepseek-raises-api-prices-and-splits-billing-into-peak-and-off-peak-windows

13. **Evaluating Bounded Autonomy in Regulated Agentic AI (RegLLM)**
    (2026-09-28)
    https://arxiv.org/abs/2609.37501v1

14. **FORCE-Bench: A Benchmark, Dataset, and Evaluation Harness for Agentic AI in Enterprise Finance**
(2026-07-11)
https://arxiv.org/abs/2607.19409v1

15. **Beyond Executable Models: The Pufibara Agent Harness and the Modelica Agent Workflow Benchmark**
(2026-08-24)
https://arxiv.org/abs/2608.23653v1

16. **Repetition In Repetition Out: Towards Understanding Neural Text Degeneration**
(NeurIPS 2023)
https://arxiv.org/abs/2310.10226v1

17. **The Curious Case of Neural Text Degeneration (Holtzman et al.)**
(ICLR 2020)
https://arxiv.org/abs/1904.09751v2