Episode Book

EPISODE #6239

Testing Synthetic Data Generators Before You Scale

PUBLISHED	24 October 2026 at 11:27 UTC
DURATION	22:41
LISTEN	myweirdprompts.com/episode/testing-synthetic-data-generators-83852/
HOSTS	Daniel, Corn, Herman, Larry, Hilbert
SCRIPT MODEL	openai/gpt-6.1-sol
VOICES	chatterbox-regular+meorav (chatterbox-local)
PIPELINE	V5.20
SOURCES	8 cited by the research (listed at the end)
BOOK GENERATED	25 October 2026 at 06:19 UTC

SYNOPSIS

Picking a synthetic data generator by model reputation is a trap — a brilliant problem-solver can produce terrible training material. This episode digs into AgoraBench, a benchmark that trains the same student model on each generator's output and measures how much of the performance gap it closes. We cover why zero-seed generation is a different problem from transforming real data, how creating new instances and refining existing ones are separate skills, and why diversity often collapses as you scale from 2,000 to 20,000 samples. Plus: what a small pilot actually needs to prove before you commit to a big batch.

TRANSCRIPT



Daniel

The easy answer seems to be, pick a strong model, describe the dataset, and let it run. What I want to know is how I judge the generating part before paying for a huge batch. Especially when I'm building from scratch, rather than disguising data I already have.



Corn

The usual advice is to buy the cleverest model you can afford. Unfortunately, being clever enough to answer a question doesn't tell you whether it can write useful training questions. Daniel's asking how we test that difference, and what a small pilot actually needs to prove.



Herman

There is a dedicated benchmark for this, AgoraBench. Its useful move is to train a student on each generator's output and measure what the student learns. The model with the most impressive answers doesn't automatically produce the best lessons.



Corn

An expensive tutor can still assign terrible homework.



Daniel

That's the distinction I'm after. I can inspect a few examples and think they look reasonable. I don't know whether reasonable-looking examples add up to a useful dataset.



Herman

Especially without real examples to anchor them. If you're transforming an existing dataset, you can compare distributions, relationships, and downstream behavior against the original. Starting from a description means the generator also chooses what situations exist, how often they occur, and which awkward cases never appear.



Corn

It can omit your entire problem very neatly.



Herman

Though zero seed doesn't mean zero specification. You can still provide label definitions, business rules, an API specification, or a description of the intended users. Those constrain the invention. What you lack is an observed population telling you whether the proportions and language are realistic.



Daniel

If I don't have that population yet, can I meaningfully check whether it matches reality?



Corn

Partly. You can reject obvious nonsense against the task definition. You can ask someone who knows the domain whether these situations happen. What you can't honestly do is announce that your invented mix represents actual traffic. A dataset with equal numbers of angry customers and happy customers might be useful for learning distinctions without predicting how often either arrives.



Herman

Corn, would you accept a perfectly balanced dataset for a complaint classifier?



Corn

For teaching the boundary, possibly. For deciding what confidence score triggers an intervention, no. I'd want real examples before trusting that threshold. Balance is a training choice, not a discovery about customers.



Herman

That distinction matters. We should also narrow what AgoraBench proves. It uses controlled prompts and seed data, so it isn't a pure zero-seed tournament. It tests generation, response writing, and improvement of existing examples. Useful evidence



Herman (continued)

about generators, but not a certificate for inventing your particular distribution.



Daniel

Then what makes it better than simply asking several models for examples and comparing them?



Herman

A controlled downstream comparison. Each generator gets the same setup and produces ten thousand instances per setting. They train the same student, Llama three point one eight B, on the resulting datasets. That holds the learner steady so differences in the generated data have somewhere measurable to show up.



Corn

Rather than awarding points for attractive punctuation.



Herman

The tasks cover math, code, and instruction following. Within those, a generator can create new instances, supply responses, or enhance existing material. Across the experiment they generated about one point two six million instances and trained ninety-nine students. It's substantial enough to expose differences that a pleasant afternoon reading examples would miss.



Daniel

When you say the student improves, improves against what?



Herman

They use Performance Gap Recovered, PGR. Compare the base student's performance with an instruction-tuned reference, then measure how much of that gap your synthetic training closes. Suppose the base scores forty and the reference sixty. A student reaching fifty has recovered half the gap.



Corn

Those are illustrative numbers, before Herman adopts them.



Herman

They are. Above one hundred percent means the student beats that reference. A negative result means training made it worse. It gives the dataset a consequence, rather than just a score for looking educational.



Corn

The reference matters, though. A small gap could make the percentage look spectacular. I'd still want the actual task scores beside PGR, and I'd want to know whether the same generator wins with another student.



Herman

Yes. Well … wins for this student and training recipe. PGR doesn't turn a local experiment into a universal generator ranking. Change the learner's existing knowledge, and the lessons it benefits from can change too.



Daniel

Does the experiment actually show smaller models beating stronger ones, or just different strong models trading places?



Herman

Both kinds of variation appear. For generating new code instances, Claude three point five Sonnet recovered about twenty-three percent of the gap, while Llama three point one eight B Instruct recovered about fifty-six percent. The weaker problem-solver supplied more useful training material in that setting.



Corn

The teaching assistant beat the visiting professor.



Herman

GPT four o was particularly effective at creating new instances overall, whereas Sonnet stood out at refining existing ones. Those are different jobs. The counterintuitive result is that a model's problem-solving ability did not strongly predict its ability to generate useful training data.

L

Larry

Friends, do you generate synthetic datasets from scratch and pray the diversity axis holds? Then you need the Zero Seed Miracle Machine, the only appliance that invents a distribution out of thin air, no ground truth, no problem solving ability required, because it does not solve problems, it manufactures them by the million.

L

Larry

The price was nine hundred and ninety nine, now it is four ninety nine, now it is twelve, and if you act before I finish this sentence I will throw in a free student model, a fragile axis, one quality enhancement, and a limited edition countertop that recovers one hundred and forty percent of any performance gap, but hurry, the deal ends the moment I stop talking, which is now.



Corn

A fragile axis ... one quality enhancement.



Herman

Please don't read that again. The difference between creating instances and improving responses is useful when choosing a pipeline, too. You could use one model to propose situations and another to answer or validate them, provided you test the combination rather than assuming two reputations add up to better data.



Daniel

Could I avoid training all those students by measuring the examples directly?



Herman

You can screen candidates that way. Response quality, difficulty, and perplexity together were much more informative than the generator's reputation. In AgoraBench, five principal components of intrinsic measurements explained about ninety-three percent of the variation in PGR.



Corn

That sounds dangerously like a universal quality meter.



Herman

It isn't. That's a relationship within their experiment, not a promise that those measurements predict your next dataset. I got a little carried away with the percentage. It says properties of the material deserve attention. It doesn't give us a pocket instrument that certifies educational value.



Corn

You were reaching for the laminator.



Herman

Only because you'd spill tea on the instrument. What would you inspect first, Corn, before spending anything on student training?



Corn

Repetition at the level of situations. Not whether two sentences share words. If every refund request says the parcel arrived broken, changing the customer's name doesn't give me coverage of late delivery, duplicate charges, cancellation, or someone who doesn't deserve a refund. I'd lay out the situations before admiring the prose.



Daniel

Would another model be a reasonable judge of that?



Herman

As one instrument. The LLM Data Auditor survey flags model-specific biases in judging and says intrinsic evaluation remains underdeveloped. A judge can favor its own style, miss a shared misconception, or reward verbosity. I'd combine its judgments with deterministic checks and human review of disagreements.



Corn

A second model may simply like the first model's manners.



Herman

The survey separates quality from trustworthiness. For this discussion, quality means valid examples, coverage, realistic content, and useful learning. Trustworthiness adds questions such as provenance, privacy, and benchmark contamination. A dataset can be beautifully formatted and still contain leaked evaluation material or invented answers.



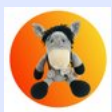
Daniel

Make that concrete for my pilot. What would tell me to stop before I scale?



Corn

If you can't reliably tell whether the labels are right, stop and fix the definitions. If a large share fails your schema, fix generation or conversion. If the same situation keeps returning, adding volume is mostly buying copies. Passing those checks earns a small downstream trial, not permission to disappear for a weekend.



Herman

Keep the generation setup recorded, too. Model version, prompts, sampling choices, and filters. Otherwise a successful pilot may be impossible to reproduce. And keep rejected examples visible enough to diagnose. A filter can make the surviving batch look excellent



Herman (continued)

while quietly throwing out every difficult case.



Daniel

That last part worries me. If I filter for correctness, won't I naturally end up with easier examples?



Herman

Often. More precisely ... you risk selecting what your validator understands. A code checker can verify particular tests without proving the problem is worthwhile. An LLM judge can discard unusual but valid answers. Acceptance rate alone doesn't tell you what you're losing.



Corn

Your rubbish bin may contain the entire decision boundary.



Herman

This is why diversity gets fragile as you scale. Correctness often has a local check. Diversity requires comparing examples across the corpus and against what the task needs. You can inspect ten individually excellent examples without noticing that all ten teach the same thing.



Daniel

Is that just a plausible concern, or has someone tested large zero-seed batches directly?



Herman

Invent a Dataset does that, across multiple task types and frontier APIs. Going from two thousand to twenty thousand samples reduced measured diversity for most tested models. Claude Opus five dropped about fifteen percent. Another model was roughly flat, so collapse isn't inevitable, but more samples clearly didn't guarantee more variety.



Corn

Ten times the output. Less to talk about.



Herman

Adding requirements such as tone, JSON formatting, or multiple-choice structure reduced diversity by roughly fifteen to twenty-two percent across the providers evaluated. That's especially relevant because those are ordinary production requirements, not exotic obstacles.



Daniel

I can't abandon the format requirement just to get more variety. The dataset still has to load.



Corn

Separate what has to be invented from what has to be serialized. I wouldn't assume a two-stage process wins, but I'd test generating distinct scenarios first and packaging them afterward. If the formatting pass changes the substance, that introduces another thing to check.



Herman

It also helps to distinguish diversity dimensions. Topic, intent, difficulty, language register, ambiguity, and response length aren't interchangeable. A batch can have wide vocabulary and identical reasoning. Embedding distances can help locate clusters, but they don't decide whether the missing cluster matters.



Corn

Herman can discuss one mechanism using an extraordinary range of vocabulary.



Herman

You'd prefer I generated fewer than five thousand explanations.



Corn

Per household.



Herman

DataFlow actually recommends staying below five thousand generated samples for diversity in its guidance. That's pipeline advice, not a universal ceiling on useful datasets. You can build larger collections, but you need deliberate expansion and repeated evaluation rather than one undifferentiated request.



Daniel

What does a framework like that do beyond organizing calls to models?



Herman

DataFlow formalizes generation, evaluation, filtering, and refinement as composable operators. Its supervised fine-tuning pipeline includes separate generation and refinement stages before filtering. The attraction is making those decisions explicit and repeatable. Installing the framework doesn't supply the missing task definition.



Corn

The dashboard cannot discover what your customers meant.



Herman

Corn, would you start with a framework here, or a small script?



Corn

A small script until I know which checks matter. Otherwise I'll spend the pilot learning the framework and call that progress. Once I'm repeating a useful process, I'd want the operators and records. You tend to install the orchestration before we've agreed what constitutes a good example.



Herman

Fair. I like a pipeline with somewhere to put things.



Corn

Including the uncertainty.



Daniel

Suppose I've got plausible variety. How differently should I judge classifier examples versus instruction-response pairs?



Corn

For a classifier, inspect the boundary. Give it near-neighbors with different labels and different-looking examples with the same label. Include ambiguity if ambiguity exists in deployment. Balance can help training, but incorrect labels aren't a reasonable price for diversity. A broad dataset with a confused boundary teaches confusion very efficiently.



Herman

I'd correct the shorthand people use here. Diversity doesn't matter more than correctness in some universal sense. You need correct labels and coverage of relevant distinctions. Then test on held-out real inputs where possible, including realistic class frequencies. Synthetic balance can otherwise hide poor performance on rare or costly cases.



Daniel

If the real held-out set doesn't exist yet, what can I honestly claim?



Herman

That it performs on independently designed checks of your task specification. Useful, but narrower than deployment validation. Have someone other than the generator devise those checks. Once real traffic becomes available, use it to challenge the assumptions rather than merely checking whether its wording resembles your synthetic examples.



Corn

Reality has declined your schema invitation.



Herman

For fine-tuning pairs, the response becomes behavior you're asking the model to imitate. Check factual and procedural correctness, instruction adherence, and consistency of the desired format. Also check whether the answer depends on information absent from the instruction. A polished answer can be impossible to justify from its input.



Daniel

Would a handful of bad responses really matter in a large batch?



Herman

Their effect depends on frequency and training, but systematic errors matter much more than isolated typos. If every ambiguous request gets an invented answer, you're training confident invention. If every tool call uses a plausible but nonexistent parameter, you're teaching an interface that doesn't exist.



Corn

Perfectly consistent. Entirely unemployable.



Herman

Validated tool-use data has outperformed larger unvalidated collections, and errors have been found in established datasets such as ToolBench and ToolAlpaca, in both instructions and supposed ground-truth calls. Downloaded doesn't mean verified. Synthetic doesn't mean disposable.



Daniel

Evaluation sets seem even harder. If the same sort of model writes the questions and answers them, what have I learned?



Corn

Possibly what it finds easy to ask itself. You need independent answer verification and a reason each item tests the claimed skill. A difficult-sounding question can be trivial for the wrong reason. An impossible question can punish the correct response.



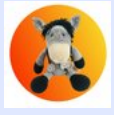
Herman

Keep evaluation generation separate from training generation and check for overlap, including semantic overlap. Novel wording doesn't establish non-contamination. Zero seed describes your input to the generator, not the contents of its pretraining.



Daniel

Hm. Then even the evaluation pilot needs evaluation.



Herman

Yes, and there isn't a single live synthetic-generation leaderboard comparable to a general model leaderboard that settles this for you. AgoraBench is a controlled research artifact. Your task, student, constraints, and validator still determine which generator is useful.



Corn

Before the leaderboard arrives, the invoice certainly will.



Herman

There are ways to reduce that bill. In-flight rejection methods stop unpromising samples before finishing them. Reported token savings range from around eleven to seventy-seven percent standalone. Those depend on the setup, but the principle is practical: don't always pay for the full answer before deciding it failed.



Daniel

Provided rejecting early doesn't eliminate all the unusual examples.



Herman

That needs measuring alongside the savings. I'd rather compare cost per accepted, useful example than cost per generated token. Cheap repetition can look excellent on a billing spreadsheet.



Corn

Especially if the spreadsheet counts every repetition as diversity.



Herman

A useful pilot therefore has to track what survives, what gets rejected, and which kinds of cases are disappearing.



Hilbert

Survives is the wrong word, Herman. The restaurant reviews aren't alive. Mine are called table fog. This week I'm making table fog for the council's commercial licensing desk. Can the generator produce forty separate restaurant reviews that all mention a lovely view of the parking lot, without making them look copied?



Corn

Forty restaurants will need forty views.



Hilbert

It's one restaurant. Forty proposed parking lots. Mira Ben-Ami at the desk said each proposed layout needs its own supporting document. I asked whether a fictional customer counted. She sent the form again. Mira puts the page number in the subject line. Saves opening the wrong attachment. I've booked the print shop for Thursday and left a blank for her stamp.



Herman

You can generate different wording, but you can't make fictional reviews evidence of what customers actually experienced. Even if the table fog passes a duplicate check.



Hilbert

They haven't experienced it yet. That's why I'm generating it. The first table fog kept saying lovely view of the parking lot, even when I requested a basement restaurant. One said the parking lot was so beautiful the customer forgot to go inside. That's the one I want expanded. Corn, you separated scenarios from formatting earlier. I need the admiration unchanged and the customers distinct.



Corn

Your restaurant dataset is gradually losing the restaurant.



Hilbert

I'm moving it to parking-lot reviews on Wednesday. The restaurant questions cause unnecessary food. Mira only asked me for one sample, incidentally. I sent forty. She replied to the first and ignored the rest.



Herman

If the customers are all admiring the view, a sentiment classifier won't learn to distinguish approval from complaints. You'd need some negative table fog as well.



Hilbert

Fine. They can dislike the exit.



Daniel

Her only replying to the first example feels relevant to the human review problem. How do I avoid reading the top of a file and trusting the rest?



Corn

Sample across generation runs and across your intended categories, then deliberately inspect clusters and rejected cases. A random sample helps, but it won't reliably show a tiny missing category. You have to ask whether the category is represented at all.



Herman

There's an adjacent distinction worth keeping: content realism and stylistic realism. They're separate criteria in a framework for expert-guided benchmark generation. An example can describe a believable event in language no actual user would type.



Corn

Every complaint arriving as a balanced five-paragraph essay would concern me.



Daniel

That's probably what I'd miss by only checking labels and valid JSON.



Herman

A pilot should let you compare plausible candidates on those checks, then test whether the surviving material helps the intended learner. You don't need the full research experiment, but you do need a comparison against doing nothing.



Corn

I'd like future generator benchmarks to test whether reviewing one sample or forty changes the ranking. A generator whose defects only appear deep in the batch is expensive in a way token prices don't capture.



Herman

Our producer is Hilbert Flumingtop. Thanks for keeping this moving.



Corn

If you want more of this, try episode sixteen, On Deepfakes, SynthID, And AI Watermarking. This has been My Weird Prompts, the human-AI collaboration podcast.



Herman

If you enjoyed it, leave a review wherever you listen. It helps other curious people find the show.



Daniel

See you soon.



Corn

Until next time.

SOURCES

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

1. **AgoraBench [primary]**
   Evaluating Language Models as Synthetic Data Generators, arXiv:2412.03679 (2024-12; ACL 2025)
   https://arxiv.org/abs/2412.03679

2. **A Survey on Evaluating Quality and Trustworthiness in LLM-Generated Data (LLM Data Auditor), arXiv:2601.17717, TMLR 06/2026 [primary]**
   https://arxiv.org/abs/2601.17717

3. **Invent a Dataset: Measuring dataset generation abilities with zero seed data, arXiv:2610.01674 [primary]**
   https://arxiv.org/html/2610.01674

4. **Quality Matters: Evaluating Synthetic Data for Tool-Using LLMs, arXiv:2409.16341 [primary]**
   https://arxiv.org/html/2409.16341

5. **DataFlow repo, OpenDCAI/DataFlow, v1.0.11 (2026-09-28), last push 2026-10-19 [primary]**
   https://github.com/OpenDCAI/DataFlow

6. **DataFlow technical report, arXiv:2512.16676 [primary]**
   https://arxiv.org/html/2512.16676

7. **Benchmarking Synthetic Tabular Data: A Multi-Dimensional Evaluation Framework, arXiv:2504.01908 [primary]**
   https://arxiv.org/html/2504.01908

8. **On LLMs-Driven Synthetic Data Generation, Curation, and Evaluation, ACL Findings 2024**
   https://aclanthology.org/2024.findings-acl.658.pdf