Episode Book

EPISODE #5981

Batch Inference: Four Doors, Half Price

PUBLISHED	15 October 2026 at 08:10 UTC
DURATION	34:14
LISTEN	myweirdprompts.com/episode/batch-inference-provider-guide-03169/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.4
BOOK GENERATED	19 October 2026 at 17:42 UTC

SYNOPSIS

Batch inference has been sitting at a 50% discount for years, and most teams still fire everything synchronously because it's easier. This episode maps the four routes into batch: first-party labs like OpenAI, Anthropic, Gemini, and Bedrock; "Flex" tiers that give you batch pricing without the file-upload plumbing; DeepSeek's off-peak pricing, which is batch economics with no batch product at all; and open-weight providers like Fireworks, Groq, Together, and DeepInfra, where the discount percentages vary wildly and the base price matters more than the discount. We also dig into OpenRouter's new batch API — one key, inline request arrays, no JSONL — and why a middleman routing layer changes the batch market from a reason to pick a provider into a commodity you route through. Plus the details almost nobody finds on their own: which discounts stack with prompt caching, why Anthropic recommends the one-hour cache TTL for batches, and why the 24-hour window is a ceiling, not a promise.

TRANSCRIPT



Corn

The same tokens. Half the price. And almost nobody using it properly.



Herman

That's the strange part, isn't it? The discount has been sitting there for years and most teams still fire everything synchronously because it's easier.



Corn

Daniel's been thinking about this one. He wants the map. Who's actually good at batch inference right now, does OpenRouter count as a batch provider, can you run batches on Hugging Face, what happens if you go local, and what are the cost-effective routes somebody might actually use.



Herman

Four doors into the same room.



Corn

And the interesting question isn't which provider. It's why this market keeps reshaping itself every eighteen months.



Herman

So before we get into who offers what, it's worth being precise about what batch actually is, because the definition has drifted.



Corn

Drifted how?



Herman

The original pitch was simple. You submit a file of requests, you give up any guarantee about when they come back, and in exchange the provider schedules your work around its own capacity and hands you a discount. That's the deal. Asynchronous, out-of-order results, twenty-four hour window, fifty percent off.



Corn

And the fifty percent is compensation for what exactly?



Herman

For giving up latency and streaming. That's it. You're not buying a worse model. You're buying the same weights with the timing taken out. The provider gets to batch your requests together, run them at three in the morning when the GPUs are idle, and keep the utilisation curve flat. You get the discount because you made their life easier.



Corn

So the discount is real because the savings are real.



Herman

The savings are real. Whether the discount is fairly priced is a different question, and we'll get there. But the mechanism is genuine. You're selling your latency back to the provider.



Corn

And the fifty percent number has become the anchor for the whole market.



Herman

OpenAI, Anthropic, Gemini, Vertex, Bedrock, Azure, Mistral, Groq, Together, Fireworks. All sitting at or near half price. It's the closest thing to a posted rate this industry has.



Corn

Except it's fraying.



Herman

It's fraying at the edges. xAI only gives twenty percent, and only on a handful of Grok models. DeepInfra gives twenty percent. Baseten gives nothing. Inference.net's published discount was only ten percent as of mid-2025. So the anchor holds for the big labs and slips for everyone else.



Corn

Which tells you the discount isn't a law of nature. It's a marketing decision.



Herman

It's a marketing decision that most of them made the same way because they're all competing for the same enterprise contracts.



Corn

So the first-party labs have all converged on basically the same plumbing, and the interesting variation is in the edges.



Herman

Nearly identical. JSONL upload, twenty-four hour target, fifty percent off, separate rate-limit pool, no streaming. OpenAI caps you at fifty thousand requests or two hundred megabytes per file. Anthropic at a hundred thousand requests or two hundred and fifty-six megabytes. Gemini takes inline requests under twenty megabytes or a JSONL file up to two gigabytes.



Corn

And the differences are in the corners.



Herman

In the corners. Anthropic keeps results for twenty-nine days. Gemini about six weeks. Azure jobs never expire, they just keep running until they finish or you cancel them. Vertex jobs can sit in queue for up to seventy-two hours before they even start.



Corn

Azure keeping them alive forever is a strange kind of kindness.



Herman

It's a quota problem dressed as a feature. They govern it with an enqueued-token quota, so you can queue as much as you want and it just sits there until the tokens free up.



Corn

What about the caching interaction? That's where the rules diverge.



Herman

They diverge badly. Anthropic's batch discount and prompt-caching discount explicitly stack, though cache hits in batch are best-effort. Vertex's implicit-cache discount explicitly does not stack with the batch discount. Same feature, opposite rules. Anthropic recommends you use the one-hour cache TTL for batches, because batches outlast the five-minute cache.



Corn

That's a detail almost nobody would find on their own.



Herman

It's the kind of thing you only learn by reading the docs twice. The one-hour TTL exists because batch jobs run long enough that the default cache has already expired by the time your second request hits.



Corn

Let's talk money for a second. What does the cheap tier actually cost?



Herman

gpt-5-nano batch is two and a half cents per million input tokens and twenty cents per million output. gpt-4o-mini batch is seven and a half cents input, thirty cents output.



Corn

Two and a half cents per million.



Herman

Rounding error. You can process a serious corpus for the price of lunch.



Corn

So the cheap tier of batch inference is absurdly cheap. That wasn't true two years ago.



Herman

The floor dropped. And the reason it dropped is that small models got good enough for classification, extraction, tagging, and reranking, which is what most batch work actually is.



Corn

Then there's the tier that didn't exist a year ago. Batch pricing without batch plumbing.



Herman

That's the new thing. OpenAI Flex, Gemini Flex, Bedrock Flex. You get the same fifty percent discount on synchronous calls. You just accept slower processing.



Corn

So you don't have to upload a file.



Herman

You don't have to upload a file. For a lot of teams that plumbing was the entire reason they never used batch. Flex removes it. You keep the synchronous endpoint, you keep the client library, you just get the discount.



Corn

And DeepSeek went further.



Herman

DeepSeek has no batch API at all. They just sell off-peak pricing. Fifty percent off every line item outside Beijing daytime peak hours, and it stacks with their cache-hit rate.



Corn

So batch economics without the batch product.



Herman

That's the whole move. They looked at when their GPUs were busy and priced accordingly. No request array, no JSONL, no job status to poll. You just call the API at the wrong time of day and pay half.



Corn

How do they know you called at the wrong time of day?



Herman

They know. Beijing daytime is the peak. Everything else is not. It's the simplest possible version of this idea and it might be the smartest one anybody's shipped.



Corn

Does it hold for the whole file, or per request?



Herman

Per request, on the timestamp when the request lands. If it crosses the peak boundary you pay the peak rate for the crossover. But in practice you'd schedule your jobs and forget about it.



Corn

Now we get to the thing that actually changed last month.



Herman

OpenRouter. They launched a batch API in late September.



Corn

Announced the twenty-second, if I remember.



Herman

The twenty-second. And the significance isn't the discount, because the discount is the same fifty percent everyone else charges. The significance is that you reach it with one API key.



Corn

So what's the plumbing look like?



Herman

No JSONL file. You post an inline array of custom_id plus body, and OpenRouter stores the JSONL itself on their end. Each batch runs on a single provider, by default the cheapest eligible batch endpoint, and you can pin it with provider.only if you want a specific one.



Corn

And the window?



Herman

Twenty-four hours, only. No choice.



Corn

Restrictions?



Herman

No streaming, no audio, no video. Images have to be public URLs, not base64. Web-search variants get rejected outright.



Corn

Results?



Herman

Deleted after thirty days.



Corn

So if you're the kind of person who leaves results sitting in a dashboard for six months, OpenRouter is not your provider.



Herman

It's not. Anthropic gives you twenty-nine days, Gemini about six weeks. OpenRouter gives you thirty days and then they're gone. Download them.



Corn

What about the performance claims?



Herman

There was a secondary write-up of the beta that reported a median completion of about seven minutes across two hundred and thirty thousand batches, with ninety percent done within an hour. That's not confirmed by OpenRouter themselves, so treat it as an anecdote.



Corn

Seven minutes median on a twenty-four hour window.



Herman

It tells you the twenty-four hour window is a ceiling, not a promise. Most small batches come back fast because there's idle capacity to schedule into.



Corn

Which is the exact opposite of what the term "batch" suggests to people.



Herman

Everyone hears "batch" and pictures a mainframe overnight job. The actual experience of submitting a batch request today is that it usually comes back before you've finished your coffee.



Corn

Unless it doesn't.



Herman

Which is why you build for twenty-four hours and get pleasantly surprised.



Corn

So why does OpenRouter matter structurally?



Herman

Because it removes the main reason to juggle separate batch accounts. You no longer need an OpenAI account, an Anthropic account, and a Together account just to comparison-shop batch across model families. One key, one request format, one job status.



Corn

That's not a feature. That's a market change.



Herman

It's a market change. The batch feature used to be a reason to pick a provider. Now it's a commodity you route through a middleman.



Corn

And the middleman sets the routing rules.



Herman

The middleman sets the routing rules, which is the part to watch.



Corn

What about their free tier?



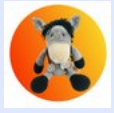
Herman

The colon-free variants are rate limited to twenty requests per minute and fifty per day. Once you've bought at least ten dollars of credit it goes to a thousand per day.



Corn

So for tiny bulk jobs it's fine, and for real batch work it isn't.



Herman

For real batch work you're paying.



Corn

Let's do the open-weight providers, because that's where the variation actually lives.



Herman

Fireworks is the interesting one. Fifty percent off serverless, plus a further fifty percent off cached tokens, and uniquely they let you pick your window. Twelve, twenty-four, forty-eight, or seventy-two hours. Input datasets up to eighty gigabytes.



Corn

So you can tell them how patient you're willing to be.



Herman

You can trade patience for priority. If you've got a week, you pick seventy-two hours and you're lower in the queue. If you need it tonight, you pick twelve.



Corn

And the gotcha?



Herman

Jobs on unsupported models can sit pending without throwing an error. You'll be waiting for nothing and no one will tell you.



Corn

Groq?



Herman

Fifty percent off, and you choose a window from twenty-four hours all the way out to seven days. But it doesn't stack with Groq's caching discount, so if you're using caching heavily the math changes.



Corn

Which models?



Herman

GPT-OSS twenty billion and hundred and twenty billion, and Llama 3.3 70B.



Corn

Together?



Herman

Up to fifty percent off on most serverless models, twenty-four hour window, OpenAI-style JSONL, fifty thousand requests or a hundred megabytes per file. Small batches often finish in minutes.



Corn

DeepInfra?



Herman

Only twenty percent off. And that looks weak until you notice their real-time base prices are already among the lowest in the market. Twenty percent off a cheap base can beat fifty percent off an expensive one.



Corn

Which is the trap with discount shopping generally.



Herman

It's the whole trap. The discount percentage is the number people quote and it's the number that matters least. What matters is the post-discount price per million tokens for the specific model you're actually going to run.



Corn

Novita?



Herman

OpenAI-compatible batch API, twenty-four hour window, fifty thousand requests or a hundred megabytes. Discount isn't stated in their docs, which tells you something about how they think about pricing.



Corn

Nebius?



Herman

Token Factory advertises batch at up to fifty percent off, aimed at very large datasets. Exact limits aren't published.



Corn

Cerebras?



Herman

Private preview via sales only, with a twenty-four hour guarantee. You have to talk to a human to get in.



Corn

Baseten?



Herman

This is the one that annoys me. They offer async inference, queue up to seventy-two hours, which sounds like batch. But there's no batch discount. You pay full rate.



Corn

So you get the latency penalty without the price benefit.



Herman

You get the latency penalty without the price benefit, and results are delivered by webhook, and if webhook delivery fails the outputs are gone. There's no queue to go back and fetch them from.



Corn

That's not a batch product. That's a queue with a tip jar.



Herman

It's an async product that people mistake for batch because the words sound similar.



Corn

Now the casualties, because this is the story underneath the story.



Herman

Lambda shut down serverless inference in September 2025 and went back to GPU rental only. Hyperbolic retired its serverless API. Kluster.ai shut down its inference business in July 2025 and pivoted to AI verification.



Corn

Three exits in about eighteen months.



Herman

And then there's the batch-first startup category specifically. Doubleword pitches itself as the Costco of inference. Realtime, Async and Batch tiers behind one OpenAI-compatible API, open models like GPT-OSS-120B, DeepSeek, Qwen, plus OCR models.



Corn

And the pricing?



Herman

Marketing says up to eighty percent off. Their own price table shows batch at roughly half of realtime. Which is the same fifty percent everyone else charges.



Corn

So the Costco comparison is doing some work there.



Herman

It's doing all of the work. And their own pages list inconsistent prices for some models, so check before you rely on them.



Corn

Sutro?



Herman

Started as a batch-inference SDK and moved up the stack into what they call AI Functions. Classifiers, judges, extractors, running on their batch engine. Platform access from five hundred a month, including a hundred dollars of inference credit.



Corn

So they stopped selling batch and started selling outcomes.



Herman

They stopped selling the commodity and started selling the thing built on top of it, which is the only rational move if the commodity is being given away by everyone else.



Corn

That's the pattern. That's the whole episode in one sentence.



Herman

It is. Batch used to be a product. Now it's a checkbox.



Corn

And the pure batch-first startups can't sell a checkbox.



Herman

They can't. Doubleword is the clearest survivor and it's still mostly selling discounted open models. Kluster died. Sutro pivoted up-stack. That's the shakeout.



Corn

I said this back when we did the batch APIs episode, and I'll say it again. The providers who own the GPUs will always be able to discount batch. The ones renting time from the providers who own the GPUs can only discount until their margin disappears.



Herman

And their margin disappeared.



Corn

Their margin disappeared about eighteen months ago. Some of them just haven't finished noticing.



Herman

That's the hosted side. The other half of Daniel's question is what happens when you don't want to use a hosted batch API at all.



Corn

Hugging Face. Can you run batches there?



Herman

Not on the router. The Inference Providers router, the OpenAI-compatible one with the cheapest and fastest routing suffixes, has no batch API at all. There's no batch endpoint to post to.



Corn

So the obvious answer is no.



Herman

The obvious answer is no and the real answer is HF Jobs. You run a Docker image or a uv Python script on Hugging Face hardware, billed per second, and you point it at a dataset or a model on the Hub.



Corn

Billed per second is the part that matters.



Herman

Default timeout is thirty minutes, so long jobs have to raise it explicitly. That's the first trap.



Corn

What's it cost?



Herman

T4 is forty cents an hour. L4 is eighty cents. A10G is a dollar. L40S is a dollar eighty. A100 eighty gig is two fifty. H200 is five dollars. Eight H200s is forty.



Corn

And no single H100?



Herman

No single H100 option. Which is an odd gap, because H100 is the workhorse for this kind of thing everywhere else.



Corn

So you either run on an A100 or you jump to an H200.



Herman

Or you rent an H100 from someone else, which is where this conversation is heading.



Corn

What does the job actually look like?



Herman

Hugging Face publishes ready-made vLLM batch recipes in the uv-scripts repo. Dataset in, dataset out, one command, any open model you name by its Hub ID. It's about as close to a one-liner as a batch job gets.



Corn

And the gotcha?



Herman

Prompts longer than the context limit are silently skipped and come back as empty strings. Unless you pass a specific flag they just don't error. You get a clean run with holes in it.



Corn

That's the worst failure mode there is.



Herman

It's the worst breaking point there is. A job that fails loudly costs you an hour. A job that fails quietly costs you a week, because you shipped the results.



Corn

What about Inference Endpoints? The dedicated ones.



Herman

H100 at four dollars fifty an hour, can scale to zero after fifteen idle minutes. But they return HTTP 502 during cold start and they have no built-in queue. You build your own.



Corn

So for batch work, Jobs beats Endpoints.



Herman

Jobs beats Endpoints by a lot. Endpoints are for serving traffic. Jobs are for chewing through a dataset once.



Corn

What's the self-hosted route look like?



Herman

vLLM offline inference is the standard. You call generate over a list of prompts and it handles the batching internally. And the interesting detail is that the vllm run-batch command accepts OpenAI Batch-format JSONL.



Corn

So the same file moves.



Herman

The same job file moves between OpenAI and your own GPU. You build the JSONL once and you can run it hosted or locally without rewriting it. That's a elegant piece of compatibility and it doesn't get talked about enough.



Corn

How does vLLM compare to SGLang?



Herman

They're close on unique prompts. About nineteen hundred tokens per second for SGLang versus eighteen fifty for vLLM on Llama 3.3 70B in FP8 on a single H100. SGLang's prefix caching helps when prompts share a long common prefix.



Corn

Which batch jobs often do.



Herman

Which batch jobs often do, if you're running the same instructions over a thousand documents. That's the whole shape of batch work. Same system prompt, different payload.



Corn

Ollama?



Herman

Not a batch engine out of the box. Parallel requests default to one, and the queue returns errors after five hundred and twelve waiting requests.



Corn

So it will fall over on you.



Herman

It will fall over on you at five hundred and thirteen. It's a local development tool that people keep trying to use as infrastructure.



Corn

llama.cpp?



Herman

The server does continuous batching by default. The constraint is consumer VRAM, which caps how many parallel slots are actually useful. On a single consumer card you might get four or eight slots and then you're memory bound.



Corn

So the local route caps out at small jobs.



Herman

The local route caps out at small jobs or overnight runs. Which is fine if that's what you have.



Corn

What does renting actually cost?



Herman

RunPod H100 is about two sixty-nine an hour on community, three forty-nine secure. RTX 4090 from thirty-four cents. Modal H100 about three ninety-five billed per second with thirty dollars a month free credit. Lambda H100 between three twenty-nine and four twenty-nine. Vast.ai interruptible at least fifty percent cheaper than on-demand but reclaimable at any moment.



Corn

So call it three fifty an hour for an H100 as a working number.



Herman

Call it three fifty and adjust.



Corn

Let's do the arithmetic, because this is the part where the standard advice falls apart.



Herman

One million documents. Five hundred input tokens each, two hundred output tokens each. That's five hundred million input tokens and two hundred million output tokens.



Corn

On gpt-5-nano batch?



Herman

Two and a half cents per million input, twenty cents per million output. Five hundred times two and a half cents is twelve dollars fifty. Two hundred times twenty cents is forty dollars. Fifty-two dollars fifty total.



Corn

On gpt-4o-mini batch?



Herman

Seven and a half cents input, thirty cents output. Thirty-seven fifty input, sixty dollars output. Ninety-seven fifty.



Corn

Now the rented H100 running Llama 3.1 8B.



Herman

At four thousand to seventy-seven hundred output tokens per second, two hundred million output tokens takes somewhere between seven and fourteen hours. At three forty-nine an hour that's twenty-five to forty-eight dollars.



Corn

So the cheapest DIY route beats the cheapest frontier batch tier by about two to one.



Herman

Two to one. And that's the number that should stop people. Everyone assumes renting a GPU is an order of magnitude cheaper than paying per token. For a small model doing a large job, it's about twice as cheap.



Corn

Twice as cheap isn't nothing.



Herman

Twice as cheap isn't nothing, and you're also running a different model. Llama 3.1 8B is not gpt-5-nano. On some tasks it's better, on some it's worse, and on structured extraction it's a different question.



Corn

Plus your time.



Herman

Seven hours of setup and babysitting is a real cost, and if your hourly rate is anything like a developer's hourly rate, a day of engineering is more than the difference on this job.



Corn

So the honest version is that renting wins clearly in specific cases, and this isn't one of them.



Herman

This isn't one of them. Renting wins when you need a specific open or fine-tuned model that no batch API serves. When prompts share a long common prefix and prefix caching actually pays. When you can use spot capacity and tolerate interruption. When data has to stay off third-party APIs for compliance reasons. Or when volume runs to billions of tokens and keeps the GPU saturated around the clock.



Corn

The saturation point is the key one.



Herman

Saturation is the whole game. A rented GPU costs the same whether it's at ninety percent utilisation or nine. Batch APIs charge you per token. So the DIY route only wins when you're keeping the GPU busy. Bursty jobs with long idle gaps lose badly.



Corn

What about privacy as a driver?



Herman

If the data can't leave your building, none of the hosted options exist and the comparison is moot. But people reach for that argument when the real reason is they don't want to do procurement.



Corn

Which is a fine reason.



Herman

Which is a fine reason, and it's cheaper than a compliance review, but let's not dress it up.



Corn

Now the pitfalls, because these apply to every route and nobody talks about them.



Herman

Results come back out of order. That's the first thing and it's the one people trip on. You get an array of results with custom_ids and they're not in submission order, and if you assumed they would be, your output is silently misaligned.



Corn

Which is the kind of bug you find three weeks later.



Herman

You find it when a downstream report looks slightly wrong. So you key on a deterministic custom_id. Hash of the input, or a stable document ID. Never a positional index.



Corn

And if you hash the input you get idempotent resubmission for free.



Herman

You get idempotent resubmission for free. Resubmit only the failures and you can't double-process anything, because the ID is the same.



Corn

What about partial failure?



Herman

A batch can report completed with individual lines errored or expired. The job status is green and the data is not complete. Most providers don't bill failed lines, so you're not paying for the failures, but you have to actually find them.



Corn

And one malformed line?



Herman

One malformed JSONL line can fail validation of the entire batch on several providers. One stray newline and fifty thousand requests get rejected.



Corn

That's brutal.



Herman

That's why you validate the file locally before you upload it. It takes ten seconds and it saves a day.



Corn

What about empty outputs?



Herman

Count them. Count the finish_reason values. A truncated output is a real failure even if the request completed successfully. A green job status does not prove the data is complete.



Corn

And structured output?



Herman

Validate every row against your schema and route parse failures to a retry queue. Use constrained decoding where the provider offers it, which cuts the parse failures dramatically.



Corn

So the practical recipe is hash your IDs, validate your file, count your outputs, and keep a retry queue.



Herman

That's the whole thing. Four habits and you skip most of the pain.



Corn

I've been thinking about Daniel's question differently while we've been talking.



Herman

How so?



Corn

He asked who's best. But you can't answer that without knowing something he didn't mention, which is how long you're willing to wait.



Herman

The window is the product.



Corn

Fireworks lets you pick twelve hours or seventy-two. Groq up to seven days. OpenRouter only gives you twenty-four and no choice. Anthropic most batches finish under an hour but the hard expiry is twenty-four.



Herman

So the window is a priority dial, and it's the thing providers actually compete on now that the price is fixed.



Corn

When everyone charges fifty percent, the only lever left is how long you're allowed to wait.



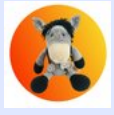
Herman

There's one more lever, which is where the job runs.



Corn

Meaning?



Herman

Whether it runs on the provider's own capacity or gets routed somewhere else. OpenRouter defaults to the cheapest eligible batch endpoint, and you can pin it. That default is doing something. It's picking the provider with the most idle capacity at that moment.



Corn

So the batch market is a capacity-clearing market.



Herman

That's what it is. The batch price is a clearing price for GPU time the provider already knows it can't sell at full rate.



Corn

And OpenRouter is the exchange.



Herman

OpenRouter is the exchange. Which is why the labs are slightly nervous about it and why everyone is watching their routing rules.



Corn

Let's bring it back to what Daniel actually asked, which is what to use.



Herman

Default case, an API batch tier. OpenAI, Anthropic, Gemini, or OpenRouter if you want open models. Fifty percent off, zero ops, done.



Corn

If you want batch pricing without async plumbing?



Herman

OpenAI Flex, Gemini Flex, Bedrock Flex, or DeepSeek off-peak. You stay on the synchronous endpoint and take the discount.



Corn

If you want flexible windows or cheap open models?



Herman

Fireworks for twelve to seventy-two hour windows, or Groq for up to seven days. Both fifty percent off.



Corn

If you need an open or fine-tuned model, or your data is already on the Hub, or your volume is huge?



Herman

HF Jobs with the vLLM uv script, or RunPod and Modal with vLLM. That's the DIY route and it's the cheapest per token if you can keep the GPU busy.



Corn

And if you're privacy-critical or the job is small?



Herman

Local. llama.cpp for a straightforward run, or a tuned Ollama overnight if you've got the patience for it.



Corn

That's the whole decision tree.



Herman

That's the whole decision tree, and it's shorter than the provider list suggests, which is the real finding.



Corn

The answer to Daniel's question is that the answer is boring. Almost everybody should just use an API batch tier and stop thinking about it.



Herman

Stop thinking about it and spend the saved engineering time on the prompt.



Corn

Which is where the actual quality comes from anyway.



Herman

Every time.



Hilbert

You're both wrong about the discount.



Hilbert

The reason batch is cheap isn't scheduling. The provider isn't rearranging its day around you. It's selling you the tokens it already decided it wasn't going to use. That's a different business.



Corn

What's the difference?



Hilbert

Scheduling means they move your job. Selling surplus means they were never going to run that capacity anyway. One of those is a service. The other is a clearance rack.



Herman

I'll take the correction. The end result is the same to the buyer, but you're right that the framing changes who's actually in control.



Hilbert

I ran a job once that finished at three in the morning. Eleven minutes. Started at two fifty. Fifty-four thousand rows.



Corn

Fifty-four thousand.



Hilbert

I'd told the client seventy-two hours.



Herman

And the client?



Hilbert

Never called back to check. I sent the file and heard nothing for nine days, and then they asked if it was still running.



Corn

Did you tell them it had finished?



Hilbert

I told them it had finished. They didn't believe me, because the quote said seventy-two hours, so I sent the log.



Herman

The timestamp was the proof.



Hilbert

Then they wanted to know why I'd quoted seventy-two hours, and I said because that was the window, and they said well yes but you didn't have to use it.



Corn

That's the whole batch problem in one exchange.



Hilbert

It is. The window is a ceiling and everybody reads it as an estimate. But here's what I actually came in to say, which is that I still have the JSONL.



Herman

From that job.



Hilbert

It's on the good drive.



Corn

The good drive.



Hilbert

It's the one that hasn't failed. The other ones are fine, mostly, but they're not the good drive.



Corn

How many drives do you have?



Hilbert

That's not the question. The question is that I'm still owed for the output tokens that came back as empty strings.



Herman

From which provider?



Hilbert

A company that doesn't exist anymore. Ran on a model that got deprecated two years ago. There were four thousand and some rows that came back empty and I was billed for every one of them.



Corn

And you've been trying to get a refund from a defunct company.



Hilbert

I have a request in. It's printed out. It's pinned to the corkboard by the desk.



Corn

The corkboard by the desk.



Hilbert

I check it every morning on the way in. It's been on the corkboard eleven months.



Herman

Eleven months for a refund on four thousand rows of output tokens.



Hilbert

It's not the money. It's the principle of the empty string. If the model didn't produce anything, I don't owe for it, and I'd say that to anyone's face.



Corn

Including a company that no longer exists.



Hilbert

Especially a company that no longer exists. They've got nothing else to do.



Herman

And the model that was deprecated two years ago?



Hilbert

Still runs on my box. Deprecated is not the same as deleted, it just means they stopped selling it. I've got the weights and I've got the JSONL, and if it comes to it I'll just rerun the four thousand rows myself.



Corn

On the good drive.



Hilbert

The file's on the good drive. The weights are on the one with the fan that rattles.



Herman

So you're one fan away from closure.



Hilbert

I've been one fan away from closure for eleven months. The refund request would settle it faster, but they've got to find someone who worked there first.



Corn

The arithmetic on the refund is what, a few dollars?



Hilbert

Four thousand and eighty-two rows, average output a hundred and ninety tokens, at the old rate. It comes to sixty-one cents.



Corn

You've had a sixty-one cent refund request pinned to a corkboard for eleven months.



Hilbert

I check it every morning. Some mornings I check it twice, if the mail's late.



Herman

Sixty-one cents, and the postage to argue about it would be more than the refund.



Hilbert

It's not about the postage.



Corn

You've been at that desk for six years and you've never once mentioned the corkboard.



Hilbert

You've never asked about the corkboard.



Corn

So the thing your story actually illustrates is that batch used to be a product and now it's a checkbox.



Herman

And the sixty-one cents is what a checkbox looks like when it goes wrong. Nobody's accountable for it because it's too small to be anybody's problem.



Corn

Which is the question we want to leave you with. The pure batch-first startup category has produced one clear survivor and two exits in about eighteen months. Doubleword is still standing. Sutro pivoted up-stack into AI Functions. Kluster shut the entire business down and went into verification.



Herman

Is that a stable equilibrium, or does the feature keep getting absorbed until there's nothing left to sell?



Corn

Everything we've covered today suggests the second. The labs absorbed the pricing. OpenRouter absorbed the routing. Flex absorbed the plumbing. What's left for a batch-first company to own?



Herman

The job file, maybe. If vLLM and OpenAI are both reading the same JSONL, then the format is the interface, and whoever owns the format owns the category.



Corn

Which is why it's worth watching whether OpenRouter's inline array format takes hold or whether JSONL stays the lingua franca.



Herman

That's the one to watch.



Corn

So the structural point to leave you with is that batch used to be a product and now it's a checkbox. The interesting question is what the next thing that becomes a checkbox is, and whether the providers that absorbed batch will absorb that too.



Herman

Because they will, if the pattern holds.



Corn

They will. Every time. That's what a maturing platform does. It eats the layer above it.



Herman

Which means the safest place to build is the layer above the layer above.



Corn

That's a good place to stop, before I start giving startup advice.



Herman

Thanks to Hilbert Flumingtop, our producer, who is currently checking a corkboard.



Corn

If this was your kind of episode, go back for episode twenty-four sixty-four, Batch APIs; episode eight, Building Your Own Whisper; and episode seventeen, Cloud Render Superpowers. This has been My Weird Prompts.



Herman

The human-AI collaboration podcast. If you want to send us your own prompt on Telegram, that's t dot me slash MWP listener bot.



Corn

And if you've built something on top of a batch API that got absorbed, we'd like to hear about it.



Herman

We'll be back soon.



Corn

See you tomorrow.