Episode Book

EPISODE #5804

Text Classifiers: Local Models vs Always-On Endpoints

PUBLISHED	9 October 2026 at 10:46 UTC
DURATION	26:22
LISTEN	myweirdprompts.com/episode/text-classifier-landscape-local-vs-saas/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.2
SOURCES	12 cited by the research (listed at the end)
BOOK GENERATED	19 October 2026 at 17:40 UTC

SYNOPSIS

Daniel has a home inventory system, a box of soldering supplies, and one question: is there a casual SaaS for text classification, or should he just run a private model locally? The answer turns out to be a landscape, not a product. Specialist classifiers beat general LLMs on both cost and accuracy — roughly twenty-five times cheaper and ahead on intent benchmarks — which makes the local instinct right and the SaaS pitch harder to justify. The features Daniel wants do exist, several times over: upload labeled pairs, retrain on a cadence, call inference over a stable endpoint. What doesn't exist is a household name for it. Meanwhile AWS Comprehend charges for a provisioned endpoint whether or not anything is flowing through it, and a six-year-old forum post asking for exactly Daniel's use case got an answer that still applies today.

TRANSCRIPT



Corn

Ten milliseconds. A specialist classifier makes a decision in about the time it takes you to blink twice, and the compute costs about five dollars and seventy cents per million messages. A general-purpose LLM doing the same job costs about a hundred and forty-two.



Herman

Which tells you your instinct about running it locally is basically right before either of us has said a word about where to actually put it.



Corn

Daniel's got a home inventory system. He keeps assets, each with an ID, and storage boxes, each with a purpose. Electronics. Electrical. And when he's holding something like a box of soldering supplies, asset ID one thousand, he wants to click one button, the name goes through a classifier, the classifier says electronics, the UI shows him the box ID, he walks over and puts it in.



Herman

And then he asked the question that ruins the fun. Is this a SaaS?



Corn

Right. His imagined version: you upload your training pairs, maybe it retrains on some schedule, and then inference is a serverless API call. Nothing running on your machine. And if two people are labeling, they can work on the same training set. He's heard of AWS Comprehend, suspects it's locked to always-on inference, and wants to know if something more casual exists. Or whether the elegant answer is a private model on Hugging Face behind their API.



Herman

So it's a landscape episode.



Corn

It's a landscape episode.



Herman

Start with what the thing actually is, because the shape of the task decides everything. A text classifier assigns a label to text. That's the whole definition. Hugging Face's own docs say it flatly: assigning a label or class to a given text. Sentiment, intent, language, topic. The label set is yours to define.



Corn

And in Daniel's case the label set is literally the names of boxes.



Herman

Which means the classifier is doing nothing conceptually harder than a very good clerk with a laminated list. Single-label. Schema-defined. Short inputs. Low volume, a handful of items a day. There's no ambiguity about what the output should look like, because he already built the box.



Corn

So the task is small enough that every product in this space supports it. Which means the question isn't capability at all. It's economics and ergonomics.



Herman

It's entirely economics and ergonomics. And the economics start with why local works so well here. A few labels, short inputs, low volume. Tuatara Decide, which is Cymetica's classifier product, runs a specialist model in about ten milliseconds and puts the compute at roughly five dollars seventy per million messages. The general LLM they compare it against is about a hundred and forty-two dollars per million. Twenty-five times.



Corn

And it's not trading accuracy for that. It's the opposite.



Herman

Their embedding-head model scores ninety-four point oh nine on BANKING77. The general model scores ninety-two point four. So the small specialist is cheaper by a factor of twenty-five and ahead on the intent benchmark. That's not a close call. That's the entire thesis of small models in one table.



Corn

Which makes Daniel's instinct right, and it also makes the SaaS half of his question harder to answer charitably. If the local version is free and wins on accuracy, what exactly is the SaaS selling?



Herman

Convenience. And it's selling something real. You don't have to run a runtime. You don't have to think about dependencies or a container or what happens when the machine reboots. You call an endpoint.



Corn

And with a couple of people collaborating, the training set isn't sitting in one person's folder.



Herman

Right. Which is the interesting part of what Daniel described. Not the inference. The training set as a shared artifact.



Corn

So walk it. What does his imagined product look like when you find it in the wild.



Herman

Upload labeled pairs. Retrain on some cadence or on demand. Call inference over a stable endpoint. And the closest match to that description is Nyckel. You upload labeled samples, it trains automatically, and it retrains automatically as new data arrives. The thing you call is a stable function endpoint.



Corn

And Nyckel's own numbers on training time are almost silly. Thirty to sixty seconds, and as few as two samples per class.



Herman

Two samples per class is not a typo. Their comparison table says other tools want hundreds to thousands. Nyckel will stand up a working model from a couple of examples per label in under a minute.



Corn

That's a demo you could run during a conversation.



Herman

It changes what the tool is for. If training is a minute, you stop treating the model as infrastructure and start treating it as a draft you revise. You add the three items it got wrong, hit retrain, and it's a new model before your coffee cools. That's the ergonomic shift. Not the accuracy. The loop time.



Corn

And pricing.



Herman

Free tier is zero dollars, a hundred invokes a month, up to five functions. Starter is a hundred and forty-nine a month, fifteen thousand invokes, up to thirty thousand samples, private inference. Business is five hundred and ninety-nine a month, three hundred



Herman (continued)

thousand invokes, decision thresholds, human review queues. Overage is half a cent per invoke on Free and Starter, a tenth of a cent on Business.



Corn

So the free tier at a hundred invokes a month. Daniel's doing a handful of items a day.



Herman

He'd fit. Barely, if he's disciplined. But it's the wrong reason to pick it, and it's a fine reason to try it.



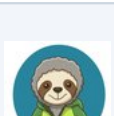
Corn

Then there's Morph. Reflexes.



Herman

Train a text classifier from labeled examples in one API call. You send the training data inline, you poll a job, and when it's done you post to the predict endpoint. A small Reflex trains in about thirty seconds. Minimums are two labels and five examples per label.



Corn

That's the whole setup.



Herman

And where it gets interesting is the retraining. It supports warm-start continual training, so you're not rebuilding from scratch every time you add a label. It can synthesize training data for you, five hundred examples per label by default, up to a thousand. And it can label your unlabeled texts, up to twenty thousand of them in a pass.



Corn

Wait. It'll label the unlabeled backlog.



Herman

You point it at the pile of things you never got around to categorizing and it proposes labels for all of them. Then you correct the ones it got wrong, and now those corrections are training data.



Corn

So the unlabeled backlog stops being debt and becomes the training set.



Herman

That's the most useful trick in the whole landscape, and it's a footnote on their docs page. The bottleneck in most labeling projects is never the model. It's that nobody wants to sit down and label three hundred items by hand. If the model proposes and the human corrects, you've flipped the job from generating labels to reviewing them.



Corn

Which is the same shape as the thing Daniel actually wants. He's not sorting items one at a time into boxes because he enjoys it.



Herman

He wants to review a suggestion and hit accept.



Corn

And Cymetica. You said Tuatara Decide runs the classifier. What about training it.



Herman

You train from JSON or CSV, text and label, and you retrain by posting more examples. The detail I'd single out is what happens during a retrain. It keeps answering with the previous version until the new one is ready.



Corn

So there's no window where the endpoint is down.



Herman

There's no window. You're never serving from an empty slot. If a retrain goes wrong you still have the old model answering, and you can look at what the new one would have done.



Corn

That's the thing you don't think about until it bites you. You retrain on a Sunday, something's off in the new data, and now every request until you notice is wrong.



Herman

Plans: Free, one classifier, four hundred thousand decisions a week. Pro at twenty dollars a month gets you three classifiers and two million a week. Max five-X is a hundred a month, Max twenty-X is two hundred. Limits are two hundred and fifty-five labels, two hundred thousand examples, two thousand characters per text, and two hundred and fifty-six texts per call. And training a classifier counts as five thousand decisions.



Corn

Five thousand decisions per training run. Which on the free tier means one retrain is about one percent of your weekly allowance.



Herman

It's a rounding error for anything that isn't retraining constantly.



Corn

And then the rest of the field.



Herman

Azure AI Language has custom text classification. You build a project, label data, define a schema, train, deploy, call it over REST. It does single-label and multi-label. Komprehend's custom classifier two point oh classifies into custom categories you can update over time. Natif dot ai has a train-your-own classifier aimed at document sorting and routing.



Corn

All of which do the same job.



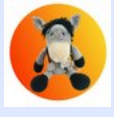
Herman

Which is the actual finding. The features Daniel described exist. Upload pairs, retrain periodically, call it serverlessly, no local runtime. That product exists, several times over. What doesn't exist is a household name for it.



Corn

So the answer to is there anything more casual than Comprehend is yes, several things. And the answer to which one is the default is nobody.



Herman

Nobody. It's fragmented. And if you went looking for the obvious choice you'd come away thinking it doesn't exist, which is the trap. It exists, it's just not one product.



Corn

So the features exist and have no owner. Now what they cost.



Herman

Which is where it stops being a feature comparison and starts being a pricing philosophy argument. Because AWS Comprehend does everything Daniel wants on the training side, and then charges you for a building.



Corn

Literally. Explain the endpoint.



Herman

Synchronous inference needs a provisioned endpoint. You buy it in Inference Units. Each unit gives you a hundred characters per second of throughput and costs five ten-thousandths of a dollar per second.



Corn

Say that in a way a human can hold.



Herman

Half a thousandth of a dollar, per second, per unit. And the billing doesn't stop when nothing's happening. The AWS pricing page says it in plain English. Charges will continue to incur from the time you start the endpoint until it is deleted, even if no documents are analyzed. That's the sentence. Everything else is arithmetic.



Corn

And AWS supplies the arithmetic themselves. Twelve hours a day of a live endpoint is twenty-one dollars and sixty cents a month in inference alone. And their own comparison says that same twenty-one sixty is what you'd pay to classify four point three million characters asynchronously.



Herman

So the real-time endpoint buys you almost nothing, for the price of millions of characters.



Corn

And the model management side is trivial by comparison. Fifty cents a month. Training is three dollars an hour, billed by the second. That part's fine.



Herman

That part is fine. It's the always-on part that isn't.



Corn

And there's a user on AWS's own forum who asked Daniel's exact question. ThomKlic. Let me get this right. You need to keep alive an endpoint all the time for just a couple of requests per day. This is way too expensive. Synchronous classification was designed for high workloads only and does not provide a cost-effective way for an infrequent amount of requests.



Herman

That's six years old and it could have been written this morning.



Corn

What did AWS say.



Herman

They offered autoscaling, time-based or utilization-based. And then said both options require you to maintain at least one Inference Unit of throughput on your endpoint, so you will continue to incur that minimum cost. The alternative is to programmatically create and delete the endpoint, which takes a few minutes.



Corn

So the official answer to I have two requests a day is either pay for one unit forever, or wait a few minutes every time.



Herman

And Comprehend Custom has no free tier at all. Training, inference, model management, all billed. There's no free path to try it.



Corn

So Comprehend is the product you'd reach for if you already had a reason to be in AWS and a volume that justifies it. For a home inventory it's a non-starter and it's a non-starter for a structural reason, not a pricing-page quibble.



Herman

It's structural. The billing model assumes you're a business with a service behind the endpoint. It can't be made cheap for two requests a day because the unit of sale is time, and time passes whether you use it or not.



Corn

Now Hugging Face. Because that was Daniel's specific guess and it deserves a fair hearing.



Herman

It deserves a very fair hearing, because the serverless side is excellent. Inference Providers is a unified API over hundreds of models. Text classification is a first-class task. You post text, you get back a label and a score. There's a free tier, and they say there's no extra markup on the provider rates.



Corn

So the API is real and it's clean.



Herman

The API is real and it's clean. And it serves existing models from the Hub. That's the catch. It is not an upload-your-training-pairs-and-we-retrain service. You're calling somebody else's classifier, not yours.



Corn

Which is fine if you want sentiment analysis. It's useless if your labels are electronics and electrical.



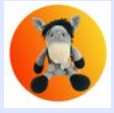
Herman

And the moment you want your own classifier served, you're into dedicated Inference Endpoints, which are billed by the hour per instance. Cheapest CPU instance is three point three cents an hour, call it twenty-four dollars a month if it's always on. Their basic two-vCPU example is forty-six seventy-two a month. GPU starts at fifty cents an hour for a T4 and goes up to five dollars an hour for an H200.



Corn

Five dollars an hour. Which is a hundred and twenty a day.



Herman

There's scale-to-zero, which sounds like the fix. But scaled-to-zero endpoints still count against your quota, and the billing model is still hourly for the instance itself.



Corn

So Hugging Face gives you the elegant call and, the moment you want your own model, hands you the same problem Comprehend has.



Herman

The same problem in nicer packaging. Elegant for calling a public classifier. For a private custom classifier it's an always-on charge just like Comprehend's.



Corn

Which means Daniel's guess is half right. The API is exactly as elegant as he suspects. It's just not serving his model.



Herman

Unless he uses the free serverless tier for a public model and accepts that it isn't his. Which, for a home inventory, would be a strange thing to accept, because the entire task is his labels.



Corn

So we have two billing philosophies and they're not compatible.



Herman

Per-hour endpoint billing versus per-request billing. AWS and Hugging Face Endpoints charge for provisioned time. Nyckel and Cymetica charge per invoke or per decision. And Nyckel says the quiet part out loud in their marketing, pitching per-request pricing against what they call confusing pay-per-hour GPU usage.



Corn

They named the enemy.



Herman

They named the enemy and the enemy is time.



Corn

And for this task, time is the wrong thing to charge for. Daniel's classifier would sit idle almost all day. It's the definition of a bursty workload. Nothing happens, then one asset needs sorting, then nothing happens for six hours.



Herman

So run the comparison honestly. On the always-on side, the cheapest option we found is the Hugging Face CPU instance at about twenty-four dollars a month, and that's before you've trained anything custom. Realistically, a private custom classifier on hourly billing is somewhere between twenty-four and forty-seven a month for CPU, and much more if you want a GPU for any reason.



Corn

On the per-request side, Cymetica's free tier is four hundred thousand decisions a week. Nyckel's free tier is a hundred invokes a month and Daniel's use fits inside it.



Herman

Which is either free or twenty dollars a month, and the twenty is Cymetica Pro with three classifiers and two million decisions a week.



Corn

So the gap isn't a factor of two. It's free versus twenty-four a month for a workload that would fit in the free tier a hundred times over.



Herman

And the local option is zero on both lines. No subscription, no endpoint, no quota. You train on the machine you already own, you call a function, it returns a label.



Corn

Which brings the argument all the way back to where Daniel started. He said his instinct was that this is a pretty perfect use case for a small model running locally, even on CPU.



Herman

His instinct was right, and the pricing pages agree with him. The general LLM comparison says a specialist is twenty-five times cheaper. The AWS thread says synchronous classification was built for high workloads. The Hugging Face endpoint pricing says a private classifier costs you something every hour it exists. All three point the same



Herman (continued)

direction.



Corn

The always-on tax is the whole story.



Herman

Every service that bills for provisioned time is structurally incapable of being cheap for a few requests a day, no matter how good the model is.



Corn

There's an honest set of unknowns here. We didn't find a single dominant casual product. The collaborative training set Daniel specifically wants, multiple people editing the same pairs, wasn't documented on any vendor page we could find. That might exist and be undocumented. Vertex AI's AutoML text classification didn't get verified at all.



Herman

The Hugging Face serverless path does not appear to offer bring-your-own-training-pairs with retraining. That's a negative finding and it's worth stating plainly, because it's the thing Daniel guessed and the guess is wrong.



Corn

The verdict for his actual use case: a local CPU model, or a per-request SaaS if he wants the training set out of his own machine. Everything else is overbuilt.



Herman

The collaborative bit is the one requirement that might push him to SaaS even though the economics don't. If two people need to edit the same training set, that's a reason to pay.



Corn

A small one.



Herman

But a real one.



Corn

I want to go back to something in the middle of all that. You said Nyckel trains in thirty to sixty seconds and needs two samples per class.



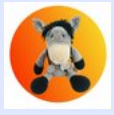
Herman

Two.



Corn

What actually happens in those thirty seconds.



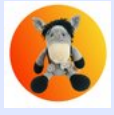
Herman

Nothing mystical. It's not doing anything like what people imagine when they hear training.



Corn

The model underneath was already built.



Herman

The model underneath was already built. Nyckel's claim of thirty to sixty seconds is a claim about the last layer, essentially. And that's the honest reason small classifiers are so tractable. You're not building a brain. You're fitting a decision surface to vectors



Herman (continued)

somebody else already computed.



Corn

Which is also why the local version is so cheap. Ten milliseconds isn't fast because of clever code.



Herman

It's fast because the arithmetic is tiny. Embedding plus a head.



Corn

This is the part that makes the whole SaaS question feel slightly off to me. If the model is a thin layer on top of an embedding, the thing you're outsourcing is not the model. It's the bookkeeping.



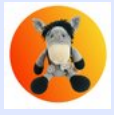
Herman

The bookkeeping and the endpoint. That's exactly what you're buying. Versioning, retraining triggers, the API surface, the concurrency.



Corn

Say that again, because I think it's the cleanest way to frame the choice.



Herman

You're not renting a classifier. You're renting somebody to keep the classifier alive.



Corn

For one person classifying a soldering kit, keeping it alive is one small process on one machine.



Herman

Which is why the free tiers exist, incidentally. The bookkeeping is cheap to provide at low volume. The expensive part is the always-on serving, and that's precisely what the per-request vendors avoid selling.



Corn

The fragmentation isn't an accident. It's what the pricing models do to the market. The per-hour vendors can only serve businesses. The per-request vendors serve everyone but there's no scale advantage to being first, so nobody becomes the default.



Herman

I don't think that's quite the whole explanation. There's also no interoperability. Your training pairs aren't portable between Nyckel and Cymetica and Morph. So the switching cost keeps you where you land.



Corn

Which is why Daniel should pick based on where he's going to want his training set in two years.



Herman

If the answer is in a folder on his own disk, the local model wins.



Corn

Let's do the cutting-room floor thing. What's the most interesting adjacent detail you couldn't fit.



Herman

Morph's label_data endpoint. Up to twenty thousand texts at a pass. It's a two-line footnote on their docs page, and it's the difference between a labeling project and an afternoon.



Corn

Twenty thousand. That's not a feature, that's an outsourcing of the worst part of the job.



Herman

It's the worst part of the job, written as an API call. And nobody markets it as the headline, which I find odd.



Corn

Where does that leave the landscape.



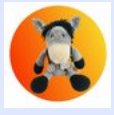
Herman

It leaves it fragmented with no household name, and honestly, that might be permanent. Every serious vendor is either selling time, which locks out low volume, or selling requests, which has no winner-take-all dynamic.



Corn

The collaborative training set Daniel wants.



Herman

That's the gap. Nobody documented it, and it's the one part of his product description that isn't commoditized.



Corn

The other thing worth sitting with. As small models get cheaper and faster on CPU, the case for SaaS at low volume gets weaker, not stronger. The always-on tax isn't a solvable problem for the clouds in this niche. It's structural.



Hilbert

It's not a tax.



Hilbert

You keep calling it a tax. It's rent. You're paying to have the thing standing there whether or not it works.



Herman

That's a fair distinction.



Hilbert

I sorted. Years of it. Incoming items onto a belt, and I put them in bins according to a card laminated to the shelf above my station.



Corn

What was on the card.



Hilbert

Categories. Thirty-one of them. I can still do all thirty-one, in order. Pipes, fittings, washers, fasteners, wire, cable, adapters, fuses... miscellaneous was number nineteen, and miscellaneous was where every argument in that building happened.



Herman

Because the category was ambiguous.



Hilbert

Because the category was honest. Everything that didn't fit somewhere did fit there, and two people could disagree about whether a thing belonged in miscellaneous or in the bin next to it, and neither of them would be wrong. I was paid by the hour. So if the belt was empty I was still there, still paid, still standing at the shelf. Same arrangement your AWS has. Provisioned, not consumed.



Corn

You were the endpoint.



Hilbert

I was the endpoint. One laminar unit of throughput, eight hours a day, no autoscaling. There was no setting on that job for less than a full shift.



Herman

When the belt was quiet, you were still paid.



Hilbert

I was still paid. That was the deal, and it was a good deal, and the company knew exactly what it was buying.



Corn

What was on the card that we haven't covered.



Hilbert

There was a category that was just a drawing of a duck.



Herman

A drawing of a duck.



Hilbert

A duck. Small, in pencil, about the size of a thumbprint. No word next to it. Nobody ever explained it. I asked twice, and both times the answer was a shrug, so I stopped asking and sorted things into it anyway. For years.



Corn

Things that were ducks?



Hilbert

Things that looked like a duck should go there. That was the rule I applied, and it held up, because nothing that looked like a duck went anywhere else.



Herman

You never found out what the category was.



Hilbert

I never found out. I still have the card. It's in a box at home.



Corn

What's the box labeled.



Hilbert

Miscellaneous.



Corn

Of course it is.



Hilbert

I've never once been able to find anything in that box. Which I take as proof the category was a mistake from the start.



Corn

That's not proof of anything.



Hilbert

It's proof for me. I put the card in there, and I know it's in there, and I can't lay hands on it.



Herman

The duck is still on it though.



Hilbert

The duck is still on it. Somewhere.



Herman

Where does that leave us. After all of that, the answer for Daniel's use case is a local CPU model, or a per-request service if he needs the shared training set, and everything else is built for a business that doesn't exist in his hallway.



Corn

Which is the right verdict, and it's a slightly funny one, because his original instinct was correct and the entire landscape tour was us confirming it. The always-on tax is structural. The clouds can't fix it without abandoning hourly billing, and they won't.



Herman

Unless the collaborative training set becomes a real product. That's the one gap where somebody could actually win.



Corn

Or the duck. Somebody could sell a classifier with a duck category and no explanation, and honestly I'd trust it more than the pricing pages.



Herman

That's the show. Hilbert Flumingtop produces it, and we're grateful he keeps the lights on back there.



Corn

Send us your own prompt on Telegram at t dot me slash MWP listener bot. We read everything, and occasionally understand some of it.



Herman

This has been My Weird Prompts.



Corn

The human-AI collaboration podcast. We'll be back soon.

SOURCES

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

1. **Amazon Comprehend Pricing [primary]**
   https://aws.amazon.com/comprehend/pricing/

2. **HF Inference Providers, Text Classification task [primary]**
   https://huggingface.co/docs/inference-providers/en/tasks/text-classification

3. **HF Inference Providers overview [primary]**
   https://huggingface.co/docs/inference-providers/en/index

4. **HF Inference Endpoints Pricing [primary]**
   https://huggingface.co/docs/inference-endpoints/en/pricing

5. **Nyckel Text Classification API [primary]**
   https://www.nyckel.com/products/text-classification-api/

6. **Nyckel Pricing [primary]**
   https://www.nyckel.com/pricing/

7. **Morph, Train a Custom Reflex [primary]**
   https://docs.morphllm.com/sdk/components/reflexes/custom

8. **Cymetica Tuatara Decide [primary]**
   https://cymetica.com/decide

9. **AWS re:Post, Sporadic real-time classification (asked ~6 years ago)**
   https://repost.aws/questions/QUwot3ttKmQLuzjWGXe5g28w/sporadic-real-time-classification-how-to-do-it-cost-efficient

10. **Azure AI Language Custom Text Classification**
    https://learn.microsoft.com/en-us/azure/ai-services/language-service/custom-text-classification/overview

11. **Komprehend Custom Classifier 2.0**
    https://komprehend.io/custom-classifier

12. **Natif.ai Train-your-own Classifier**
    https://developer.natif.ai/create_custom_workflows/train_your_own_classifier/