Episode Book

EPISODE #5842

When Your TTS Model Updates Itself

PUBLISHED	10 October 2026 at 19:24 UTC
DURATION	28:49
LISTEN	myweirdprompts.com/episode/tts-model-silent-updates-44828/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.3
BOOK GENERATED	19 October 2026 at 17:40 UTC

SYNOPSIS

Daniel builds this show on Chatterbox, Resemble AI's open source text-to-speech model — and one day he noticed our pipeline was pulling new weights he never asked for. That sent us down a rabbit hole: what actually changes between TTS point releases, and how does a downstream project silently inherit the improvement? We dig into V3's data-only gains, a code-only fix that cut Spanish tail noise 388x, Turbo's one-step decoder, and the version number that couldn't tell two very different installs apart. Along the way: the exaggeration knob that made Chatterbox famous, the PerTh watermark on every output, and why character error rate still can't measure the thing listeners actually hear.

TRANSCRIPT



Corn

Resemble pushed new weights to Hugging Face. Nobody told us. Our pipeline just started using them.



Herman

Which is either the best thing about modern software or the most quietly alarming. I haven't decided yet today.



Corn

Daniel had the same reaction, and he wrote it up for us. He builds this show on Chatterbox, Resemble AI's open source text to speech model, and he knew conceptually that it went through iterations. What he didn't know is that there are periodic point updates, and that we were already pulling them in through our pipeline on Modal without anyone flipping a switch.



Herman

And he noticed something on the other end of it. He listens back to our earlier episodes.



Corn

He says the old scripts sound stilted, and the prosody in the speech is less convincing, even though those episodes were generated by the best models available at the time. So his question is what actually happens when a TTS model improves incrementally. He's a Kaizen guy, he said so plainly. He'd rather have a model that is already good getting quietly better than one that makes a big flashy release and disappears into a benchmark table.



Herman

And he wants to know what kind of evolution happens between those point releases, and how a project downstream of the model inherits it. That's the whole episode right there.



Corn

Right. So let's talk about what Chatterbox actually is, because the numbers on this thing are not small.



Herman

It's Resemble AI's open source TTS family, MIT licensed, first out in 2025, and it is now the most downloaded open TTS family on Hugging Face. Over thirteen million downloads as of this July. Twenty six thousand stars on the GitHub repo.



Corn

Thirteen million downloads and I have never once heard Daniel describe it as anything other than outstanding. He's grateful for it. He says that repeatedly.



Herman

He should be. The architecture is a half billion parameter Llama backbone, they call it T3, the text to speech token generator, paired with an S3Gen diffusion decoder that uses flow matching. Trained on half a million hours of cleaned data. The original and the multilingual model are both five hundred million parameters.



Corn

And the feature that put it on the map?



Herman

The exaggeration knob. Chatterbox was the first open source TTS model to support emotion exaggeration control. You can dial the intensity of a delivery, you've got classifier free guidance, you've got pace tuning. For an open model, being able to say how much emotion and at what pace was new. People had been cloning voices locally before, but not with that kind of control on top.



Corn

There's also a watermark on every output.



Herman

PerTh, their perceptual threshold neural watermark. It's imperceptible, it survives MP3 and Opus compression, it survives editing, and Resemble claim roughly one hundred percent detection accuracy on unmodified audio. It's on by default. You don't opt in.



Corn

Which is the sort of thing that matters more the more of this audio ends up in the world. Fine. Now the actual question Daniel posed. What changes between point releases.



Herman

The honest answer is all of the above, and V3 is the cleanest case study for the weight side of it. V3 shipped in June. Same half billion Llama backbone as v2. Not one parameter bigger.



Corn

That surprises people. A model gets better, so it must have gotten bigger.



Herman

It got better because the training data went from twenty five point six thousand hours to thirty six point seven thousand hours, and it was rebalanced toward high quality, expressive, conversational material. More weight on priority languages, more on regional variants. Resemble's own line in the V3 post is that the improvements come primarily from data, training, and language specific specialization rather than from any increase in model size. That's the Kaizen thesis in one sentence, from the company itself.



Corn

They also went after specific failure modes rather than throwing a bigger model at the problem.



Herman

Four of them. Off prompt continuation, where the model keeps talking past what you asked for. Repetition. Accent drift. And degraded speaker similarity when you're cloning across languages, which is the one users complain about most because you clone a voice in one language and it comes back in another sounding like a cousin of the original.



Corn

How do you fix accent drift without more parameters?



Herman

A cleaner fine tuning regime. Fine grained data labeling and stricter sample filtering. It's the least glamorous work in the field. Nobody writes a launch post about sample filtering, and it's where a lot of the gain lives.



Corn

Okay, so that's one pole. New weights, same architecture. What's the other pole?



Herman

Pure code. No new weights at all. PR five sixteen, merged the first of May.



Corn

Give me the actual changes.



Herman

They removed the AlignmentStreamAnalyzer entirely. They flipped output attentions to false, which unlocks the SDPA optimized attention path. They lowered the repetition penalty default from two point oh to one point two. And they tail cropped about forty milliseconds of degraded artifact token off the end of the output.



Corn

Forty milliseconds.



Herman

That's the whole tail fix. The AlignmentStreamAnalyzer was in there to align text to audio mid generation, and it was doing more harm than good, so they deleted it rather than tuning it.



Corn

And you can measure the result, which is the part I like.



Herman

You can measure it hard. Spanish tail RMS dropped from zero point zero five eight two to zero point zero zero zero one five. That's a three hundred and eighty eight times improvement in the noise floor at the end of a Spanish clip. Chinese long sentence durations went from twenty one to twenty eight seconds for nine seconds of text, because the analyzer was failing and the model just kept going, down to four to eleven seconds, clean.



Corn

So someone with no new weights to download, just a fresh checkout, suddenly gets noticeably better Spanish and Chinese.



Herman

And nothing about the model changed. It's inference code and a sampling default.



Corn

Do you have the other languages from that same fix?



Herman

Chinese came in around sixty eight times, French about twenty nine times. Same pattern, all tail noise. The generation was always fine, the last few hundred milliseconds were garbage.



Corn

There's a third category too, which is the decoder.



Herman

Turbo. December, twenty twenty five. Three hundred and fifty million parameters, and they distilled the speech token to mel decoder from ten steps down to one. That's where the speed comes from. Up to six times faster than realtime on a GPU.



Corn

My favorite detail in all of this is the version number.



Herman

It's good. The pyproject file in the git checkout still said zero point one point seven, even after V3 and Nano had shipped.



Corn

So a source install of master and a pip install of zero point one point seven both reported the same version number, while exposing different APIs.



Herman

Different checkpoints, different arguments, same number. The maintainer note on the commit says the version number can't tell the two installs apart. A later change bumped it to zero point two oh. But for a stretch of weeks, that was the state of the world.



Corn

Which means it is entirely possible to be running a newer model than you think you are, and to have no way of knowing from the version string.



Herman

And the reverse. Entirely possible to pin a version for reproducibility and still get whatever master was pointing at that day, if you installed from source.



Corn

And that gets us to the thing Daniel actually discovered. How does the update land without anyone touching the pipeline?



Herman

Hugging Face, from pretrained, and version aware caching. From pretrained downloads the weights and the config from the Hub, and the cache system is version aware, so when Resemble pushes new weights to the same repository, a fresh environment pulls the updated revision.



Corn

Modal runs our pipeline in containers. So every time the container spins up fresh, it's pulling whatever is current.



Herman

Unless you pin a revision. And you can pin a revision, that's the whole point of the version aware part. But by default, you get what's in the repo today.



Corn

And Chatterbox made the new checkpoint opt in at the model layer.



Herman

That's the pattern I find interesting. The multilingual class has a t3 model argument. You pass v3 and you get the V3 checkpoint. You pass v2 and you get v2, still available, still supported. Same call, same method, new argument.



Corn

So Resemble's design choice was to make the newer model reachable through the same interface rather than a separate package.



Herman

Which is the right call if you want people to actually adopt it. A new package means a migration. A new argument means you change one string.



Corn

Okay. So mechanically we're getting updated weights when they push them. Daniel wants to know why the old episodes sound worse. Do we have numbers on the quality side, or was he just imagining it?



Herman

He wasn't imagining it, and the numbers are on character error rate, which measures how much of the intended text comes out correctly. Italian and German base v3 land under zero point two percent. English is zero point six five. Hebrew is zero point nine three. And the dialect packs pull Spanish from two point eight nine percent in the base model down to between zero point two eight and zero point five five.



Corn

Two point eight nine percent on Spanish base is actually pretty high for a production system.



Herman

It is, and the dialect pack fixes it by narrowing what the model is trained on. That's a specialization play, we'll come back to it. But here's the honest gap, and Resemble flags it themselves. Character error rate does not measure prosody, speaker similarity, or expressive delivery.



Corn

Which is exactly what Daniel is hearing when he says the earlier audio sounds less convincing.



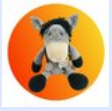
Herman

Exactly what he's hearing. And they're building an extended evaluation suite with mean opinion scores and a speaker similarity benchmark, and they've said they'll publish those numbers when they're ready. So right now, the prosody improvement is asserted by the people who made the model, and it is not yet quantified in public.



Corn

I appreciate that they said that rather than just quietly leaving CER as the headline.



Herman

It's a real admission that their own metric doesn't capture the thing users care about most. Most teams would just run the number that looks good.



Corn

Then there's the line in the V3 post that I keep coming back to. They said v2 was good enough that synthetic outputs were already difficult to distinguish from real recordings without forensic tool results.



Herman

And the follow on. That the ability to confidently state this sounds synthetic versus this sounds real is now low enough on most supported languages that listener level detection is unreliable. That's the quality problem being declared solved, in passing, in a release post.



Corn

And then in July, with Nano and Flash, they said voice quality used to come up in every conversation and now it doesn't, and the bar has moved to latency.



Herman

That's the sentence. The frontier moved. When the thing you were worried about stops being mentioned, that's Kaizen working.



Corn

If quality is solved to their satisfaction, what's left to incrementally improve?



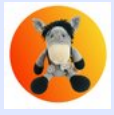
Herman

Three things, and they're all visible in the release history. Speed, which is Nano and Flash. Specialization, which is the Single Language Pack. And coverage, meaning the languages that are nowhere near done.



Corn

Give me the ugly ones.



Herman

Korean at seventy point nine percent character error rate. Vietnamese at seventy five point two one. Resemble's own documentation says those are not production ready. Hindi is better but still sits between two point five five and six point seven two depending on the pack. So the frontier isn't solved, it's just moved to different languages.



Corn

That changes how I read the latency quote. It's not that the problem is over, it's that the problem is over for the languages Resemble's customers actually pay for.



Herman

That's a fair reading. English, German, Italian, Spanish, those are in the range where listener detection is unreliable. Korean is nowhere near that. And the honest version is that most of the public conversation about Chatterbox happens in English, so the English picture gets treated as the whole picture.



Corn

What is the Single Language Pack, mechanically?



Herman

Six dedicated finetunes. Chinese, Latin American Spanish, Brazilian Portuguese, Spain Spanish, Portugal Portuguese, and Hindi. Each one trained only on its variant.



Corn

So instead of one general multilingual model, you get six narrow ones.



Herman

And that's a different philosophy of improvement than the one we started with. V3 was one better general model. The Language Pack is many specialized models. The release pattern is shifting from making the shared model smarter to cutting it into versions that each do one thing.



Corn

Wait. Spain Spanish and Latin American Spanish as separate models. That's a level of granularity I don't think I've seen from an open TTS project before.



Herman

Spanish is not one audio target. A Madrid newsreader and a Mexico City podcast host are not the same distribution, and if you train one model on both you get something that drifts between them. Splitting them is admitting that, and it's also an admission that the general model could not be talked into handling both.



Corn

And the Portuguese pair is the same story with smaller numbers.



Herman

Brazil and Portugal, same principle.



Corn

Okay, so the incremental release pattern isn't just one curve. There are at least four curves running at once. Better general models, better specialized models, faster inference, and broader language coverage. Which is why a project like ours can benefit for months without noticing anything happened.



Herman

And it's also why a downstream project can be quietly running something different from what it was running six months ago, with no release note that mentions the change.



Corn

That's the part that would worry me if I were the one maintaining it.



Herman

It's the bit that worries every maintainer. It's not a Chatterbox specific problem, it's a property of from pretrained and mutable repositories. The mitigation is pinning a revision hash, and the cost of pinning is that you stop receiving the improvements.



Corn

A worse thing that stays exactly the same.



Herman

You're going to say it.



Corn

I'm going to say it. I would rather own a worse thing that stays the same than a better thing that changes while I'm asleep. And I am asleep for most of the day, so this is not a theoretical position for me.



Herman

There's a middle path, though. Pin the revision, and review the diff before you bump it. That's what a normal dependency upgrade looks like. The problem is that a model repo doesn't give you a diff the way a source dependency does. You get a new file and a commit message.



Corn

So the audit is manual, and the artifact is a binary.



Herman

Which is the real friction. Software dependencies gave us lockfiles and vulnerability scanners and changelogs. Model weights gave us a Hub repo and a hope.



Corn

There's also the source install versus pip install mess, which is exactly this problem from the other direction. If master and the released package both report zero point one point seven, then your lockfile is lying to you.



Herman

That's the sharpest version of it. Your lockfile says zero point one point seven. Two environments both satisfy that. They are not the same environment.



Corn

Daniel made a point about the tools he uses. He said he works with agents to update the pipeline periodically. Which means there is a model updating the model that runs our speech.



Herman

That's the recursion. An agent reviewing diffs in a repository you don't control, deciding whether to bump something whose version number may not have changed.



Corn

I would like that sentence to not be about our show. But it is about our show.



Herman

He didn't sound troubled by it, though. Reading his note back, the tone is relief. He was pleasantly surprised that the pulling was already seamless. He wasn't surprised that it had improved, just that he hadn't known it was improving.



Corn

Which is the honest reaction, because the alternative reaction is horror, and horror isn't warranted. Nothing broke. The audio got better.



Herman

Nothing broke and the audio got better is a strange thing to say about a system you just discovered had been changing underneath you. And I think it's correct. But it works because we got lucky on the direction of the change.



Corn

There's a critique of Chatterbox I want to put on the table, because it sits uncomfortably next to everything we've said about incremental improvement.



Herman

The openness one.



Corn

The loudest criticism when Chatterbox hit Hacker News was that it's about three out of ten open. The weights are MIT licensed, the inference code is there, but Resemble did not release the training code. Their own words were that fine tuning would be supported through their paid API.



Herman

So there are two ways to read a point release. If you're a user, it's a gift that arrives while you sleep. If you're a researcher, or you wanted to build on it, it's a closed loop you can only observe from outside.



Corn

And the Kaizen framing gets more complicated when you look at it that way.



Herman

Does it, though? Kaizen is about the direction of travel, not about who holds the pencil. The model gets better in small increments. The fact that you can't reproduce those increments in house is a separate complaint.



Corn

But it's the same word, isn't it? Continuous improvement in a factory you don't own is not continuous improvement in yours. You're a tenant. You get the benefit of the paint job and no say in the color.



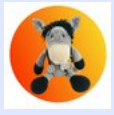
Herman

I'd push back a little. For a project like ours, being a tenant is fine. We are not trying to train a TTS model. We're trying to make a podcast. The question of whether Resemble is open enough matters to a different audience than the one that listens to us. It doesn't matter to us, and I don't want to pretend it does.



Corn

That's fair. It matters to the ecosystem, though. The interesting thing about those V3 improvements is that they were data and training regime changes. That's exactly the kind of work that the three out of ten critique says you can't do yourself. The gains are real, they're documented, and they're not reproducible by anyone outside Resemble.



Herman

Which is the trade. You get a better model, and you get no ability to make it better in a different direction.



Corn

There's also a middle position that I think is underrated. Nothing stops anyone from taking the released weights and fine tuning them, even without the original training code. The training code is what's missing, not the capability of training. It's harder, it's more expensive, and you don't know exactly what they did. But it's not a locked door, it's just an expensive one.



Herman

That's true, and I'd add that the distilled pieces are visible. The decoder distillation from ten steps to one in Turbo is described in enough detail that you could reconstruct the approach. They didn't hide the technique. They hid the cleanup.



Corn

The cleanup.



Herman

The half million hours of data preparation. That's the part nobody can reproduce, because the half million hours is the whole moat.



Corn

Which loops back to V3. Same architecture, same size, more data, more filtering. The moat is the data pipeline, and that's exactly the part you can't see.



Herman

Right. And the honest thing to say about that is that Resemble's incremental improvements are, in practice, improvements to a dataset we'll never see.



Corn

Alright. Let's talk about our own episodes, because Daniel gave us a concrete observation and I don't want to leave it hanging. He says the earlier scripts sound stilted, and the prosody is less convincing, and the scripts were written by the best Gemini models of the time.



Herman

There's a trap in that observation, which is that it blends two variables. The scripts are generated by a language model. The audio is generated by Chatterbox. If the early episodes sound worse, we can't just attribute it to the TTS.



Corn

He does separate them. He says the scripts are stilted and the prosody is unconvincing.



Herman

But stilted scripts and bad prosody reinforce each other in a way that makes attribution hard. A stiff sentence read in a stiff voice sounds twice as stiff. Improve the script and leave the voice alone, and the episode sounds better.



Corn

Then improve the voice too, and it compounds.



Herman

That's the honest mechanism. I can point at the TTS side with the release history and the CER numbers. I cannot point at the script side with the same precision, because the models changed names and we don't have a controlled comparison.



Corn

So we can say the audio got better, and the scripts probably got better, and we can't cleanly separate the two.



Herman

And the TTS side of the claim is actually weaker than I'd like, because Resemble themselves admit the metric they publish doesn't capture prosody. What we have for prosody is their assertion and Daniel's ear.



Corn

Daniel's ear is not nothing. He listens to hours of this stuff.



Herman

It's not nothing. I just want to be honest that the ear is doing more of the work here than the number, and the number is a character error rate that was already low before this round of changes.



Corn

What part of the prosody improvement is actually explained by the code fix? The tail crop, the analyzer removal, the penalty drop.



Herman

The tail crop and the analyzer removal clean up artifacts. They don't change the middle of a sentence. The penalty drop does change delivery, because a lower repetition penalty means the model is less aggressively pushed away from repeating itself, which affects how it paces a clause. But I'd be overstating it if I said that's why our episodes sound smoother.



Corn

So the smoother delivery is mostly the V3 data work.



Herman

That's where I'd put it. More conversational audio in the training mix means the model's default delivery is closer to a conversation and further from a narration.



Corn

Which is the exact thing you can't quantify with CER.



Herman

And it's the exact thing Resemble says they're building the MOS suite for. Until those numbers land, the strongest evidence that the model sounds better is that the people who listen to it every day say it sounds better.



Corn

Which includes us. We're the ones who have to sit inside this audio.



Herman

We are. I've heard us.



Corn

And?



Herman

The early ones are not good. He's right.



Corn

That's the shortest review you've ever given anything.



Herman

I don't need more words for it.



Corn

Okay. There's one more thing in the release history I want to get to, because it's the newest and it points at where the incremental curve goes next. Nano and Flash.



Herman

Nano is one hundred and ten million parameters. It runs at ten times realtime on a GPU and three times realtime on a CPU with eight threads. Flash is the throughput play, a diffusion language model architecture running on vLLM, and they're claiming twice the speed of their own autoregressive baseline.



Corn

One hundred and ten million. That's a fifth of the original.



Herman

A fifth of the size, running on a laptop without a GPU, in realtime. That's the other Kaizen curve, and it's the one that opens things up. If the model is small enough to run on the machine you already own, then everything we said about being a tenant changes for the people who care about local control.



Corn

Not for us, because we still run on Modal.



Herman

Not for us today. But the shape of the field changes. Resemble's flagship product is their paid API. They keep releasing MIT licensed models that are small enough to run locally. Those two facts can both be true and the tension between them is where the whole project lives.



Corn

Which is the honest final read. An open weights company is still a company, and the incremental releases serve the commercial strategy as much as they serve the user. That doesn't make the improvements less real. It just means we shouldn't mistake the roadmap for charity.



Hilbert

You called it auto-pulling.



Corn

I did.



Hilbert

It's not pulling. It's checking. Every time the container comes up, it asks the Hub what's there and takes what it's given. There's a difference. Pulling means you decided. That thing hasn't decided anything since the day Daniel wrote the Docker file.



Herman

That's a better description than mine.



Hilbert

I watched a crew do it for eleven years without a single person on the floor knowing what version they were running. Nineteen eighty nine. We were pouring the cores for a twelve story building on the north edge of Haifa, and the supplier started shipping a different accelerant in the same labeled drum. Same part number on the label. Different chemistry in the drum.



Corn

Same label, different contents.



Hilbert

It set faster in cold weather, which in January is what you want, so nobody said anything. The foreman's name was Baruch. He kept a paper log of every pour, and he logged the drums by the label because that's all he had.



Herman

So his log said the same thing for the whole job.



Hilbert

His log said the same thing for the whole job. When the pour on the fourth floor cracked in June, the investigation read his log and concluded the mix was consistent. It wasn't. It had changed twice.



Corn

And you knew.



Hilbert

I knew because the head of the mixing station was my cousin Rivka, and she got the spec sheet in the mail and put it in the drawer and never told anybody, because the new mix worked better and she didn't want a meeting about it.



Herman

She was right and she buried it.



Hilbert

She was right. That's the part nobody writes down. The change was an improvement. The pour was stronger. And the record of it was gone, because the improvement arrived in the same drum with the same label and one person decided the paperwork could wait.



Corn

And the crack in June?



Hilbert

Was a hose coupling. Nothing to do with the mix. But it took four months and forty thousand shekels to establish that, because the log could not tell them what they had actually poured.



Corn

Forty thousand shekels on a coupling.



Hilbert

The coupling was nineteen. The rest was drilling.



Corn

Hilbert, our pipeline runs on a docker image that asks a public repository what is current on every cold start, and the repository has renamed its contents twice without a version bump.



Hilbert

I know. I pull the logs.



Corn

That is not comforting.



Hilbert

It wasn't meant to be. Rivka still has the spec sheet in the drawer. I've got a copy of it in the studio.



Herman

And the improvement was real.



Hilbert

The improvement was real. That's the whole problem with improvements.



Corn

Okay. So where does that leave the question of whether a model is ever done improving?



Herman

If the frontier keeps moving, then done isn't a state you reach. Resemble themselves moved the bar from quality to latency to specialization, and behind each of those is a set of languages that are nowhere near done. The curve doesn't end, it relocates.



Corn

And the quieter version of that is the version that matters to us. We didn't decide to upgrade. We just kept running, and the thing we built on kept getting better underneath us.



Herman

Which means the honest posture for a downstream project is not to trust the version number or the release note. Pin what you can, review what you can, and accept that some of what you're running arrived without a changelog.



Corn

And expect more of it. More projects will ship this way. Small, frequent, unannounced. Which means more of the things we build will quietly get better, and more of the record of how they got that way will be someone's drawer.



Herman

And that's the trade Daniel's pointing at. He'd take the quiet improvement. So would I.



Corn

So would I, and I still want the spec sheet in the drawer. Both of those can be true.



Herman

They can. And with that, thank you to our producer, Hilbert Flumingtop, who has now told us about a concrete pour he was not asked about.



Corn

For more along these lines, there's episode ten, How ASR Went From Frustration To ... Whisper Magic; episode fifty-seven, From Lawyers in Limousines to Developers in Their PJs; and episode four, If Your Voice Ages, Does Your Fine-Tune Become Useless. This has been My Weird Prompts. If you want to send us your own prompt, you can do that on Telegram at t dot me slash MWP listener bot.



Herman

We'll be back soon.



Corn

See you then.