Episode Book

EPISODE #6218

Training a LoRA to Write Like You

PUBLISHED	23 October 2026 at 08:35 UTC
DURATION	21:40
LISTEN	myweirdprompts.com/episode/lora-writing-voice-clone-16819/
HOSTS	Corn, Herman, Ivan, Hilbert
SCRIPT MODEL	anthropic/claude-sonnet-5.5
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.20
BOOK GENERATED	24 October 2026 at 07:26 UTC

SYNOPSIS

Off-the-shelf "Humanizer" models flatten everyone into the same average voice — the method works, but the corpus is the problem. This episode walks through building a personal style adapter: manufacturing flat/neutral training pairs from your own posts, choosing a base model over an instruct one, curating hard, and knowing when it's overfit. Then we cover the practical part: running the adapter with llama.cpp, vLLM, or Ollama, and why serverless LoRA hosting is mostly gone.

TRANSCRIPT



Corn

Daniel tried a Humanizer model off Hugging Face today, fed it a draft, and got back text where every argument survived and every trace of him was gone. Which is a strange failure, because he used to write for a living. He knows what his own sentences sound like, and this wasn't them.



Herman

It's a good failure to look at, because the model he tried is probably doing the right thing. The most-downloaded one right now is a fine-tune of Gemma four twelve billion, and it was trained exactly the way Daniel wants to train his: AI-written draft in, human original out. Twenty-eight thousand pairs of that.



Corn

So he's reinventing something that already exists.



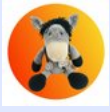
Herman

He's reinventing the half that works. Those twenty-eight thousand pairs come from paper abstracts, government reports, student essays, Reddit, Hacker News. Hundreds of authors. The model learns what the average human sounds like, and the average human is nobody. The method is fine. The corpus is the problem.



Corn

So the answer to "which Humanizer should I use" is none of them.



Herman

Right, and the answer to "what should I train" is a smaller, much more stubborn version of the same idea. Pairs built from Daniel's own writing, in one direction only: flat, neutral version goes in, his version comes out. We've done the mechanics of LoRA, the pair counts, and the fact that plain supervised training beat small-data preference tuning for voice, so I'll skip those and go to what's left. Where do the pairs come from, which base model, how do you know it worked, and how do you actually run the thing.



Corn

Pairs first. Because he has the human side already. Years of posts. What he doesn't have is the robotic side.



Herman

Which you manufacture. Take a post, cut it into chunks, a few paragraphs each, and ask a frontier model to rewrite every chunk as flat, neutral prose. Same claims, same names, same numbers, nothing added, nothing dropped, no flourishes. That rewrite is the input. His original is the target. The old style-transfer work did it this way too: paraphrase the style away, then train a model to put it back.



Corn

And the model learns what, exactly? That's where I get nervous. It sees a flat paragraph and a lively one. It could learn "add adjectives".



Herman

It can't learn only that, because the difference between flat and him isn't adjectives. It's where he puts the weight in a sentence, how long he lets a paragraph run before a short one, which word he reaches for when two would do. The model has to find that from a few hundred examples, and it can, because you are asking it for a narrow thing. It's not generating ideas. The ideas arrive in the input.



Corn

I'd still worry about the neutraliser leaving fingerprints.



Herman

You should, and it's the part people skip. Whatever model flattens your text leaves its own habits in the flat version. If you train on one neutraliser and then feed the adapter a draft from a different model, the input looks different from anything it saw. So use two or three different neutralisers with different instructions. And make the inputs vary in kind, not only in wording. Some are the flat rewrite, some are a bullet outline of the chunk, and some are a fake dictation, with the false starts and the filler words added back in.



Corn

Fake dictation. Because he dictates.



Herman

Constantly, from what we hear. Dictation comes out in the order he thought of things, not the order he'd publish them, so a third of the inputs should look like that. And if he has any real transcripts next to the posts he made from them, those are the best pairs in the whole set. Gold. I haven't seen anyone publish a set like that, so that part is my suggestion rather than established practice.



Corn

How many pairs before it stops being a toy?



Herman

People have tuned on about a hundred posts and heard a difference. My guess for Daniel, who has a lot more than that, is five hundred to fifteen hundred chunks, and the first thing I'd spend effort on is subtraction. Throw out anything an editor heavily touched. Throw out anything he wrote with AI help. Throw out the old pieces if his voice has moved. The work on turning existing text into training pairs found the same thing: raw, uncurated pairs trained badly, curated ones trained well.



Corn

Herman, you're about to say "curation is the whole game".



Herman

I was going to say it's most of it. And the flattening step needs its own checking. Rewrites by a model drop or distort content more often than you'd hope. So do the cheap automated pass first: every number and proper name in the original has to appear in the flat version, and the flat version can't be wildly shorter or longer. Then read fifty pairs by hand. Fifty. Not skim, read.



Corn

Fifty pairs is an evening. I'd lose the evening and sulk about it.



Herman

You'd sulk, and the adapter would be better for it. Here's a real question for you, Corn. When you write the flattening instruction, do you make it strict, keep every sentence, change only the words, or loose, say the same thing plainly in your own way?



Corn

Strict. If it's loose, the flat version moves the ideas around, and then the adapter learns that Daniel reorders things. He doesn't. He reorders things when he edits, and that's his decision, not the model's.



Herman

That's my answer too, and I hadn't put it that way. Strict on content, and the model is free to fix everything else, which is where the voice lives.



Corn

Next, though. The base model.



Herman

Gemma four, twelve billion, the base checkpoint, not the instruct one. That's my recommendation and I'll defend it.



Corn

The instruct one comes with a chat template, the library defaults assume it, and the training guides all say start there. Why would he make his life harder?



Herman

Because the instruct model is the one that was tuned to be agreeable. There's a paper on exactly this, base models beating aligned models at randomness and at creative writing, and the aligned ones prefer what's pleasant over what's original. Another group measured the slop directly. Some of the stock phrases show up a thousand times more often in model output than in human writing. Preference tuning is where a lot of that comes from. When you train a LoRA on top of an instruct model, you are asking a small adapter to push uphill against all of that.



Corn

A small adapter against a very big habit. So the base model has less to push against.



Herman

It hasn't been taught to be nice yet. It's a model that has read an enormous amount of human text and has no opinion about how to answer. The price is that it has no chat format, so you give it one. A plain layout, the flat text under one marker, his version under another, and you compute the loss only on his version. Twenty lines of setup.



Corn

And if the base model turns out to ramble?



Herman

Then you've lost an afternoon. Train the same pairs on the instruct model as well and compare outputs blind. I'm betting the base wins on voice. Corn, what's your bet?



Corn

Instruct, honestly. Sloths pick the thing with fewer ways to go wrong. But I've never had my mind changed by a bet, only by an output.



Herman

Fair. Run both.



Corn

Why twelve billion, though, and not the thirty-one?



Herman

Because he's going to iterate, and iteration speed beats raw size for this job. Twelve billion with quantised training should fit on a card with about ten gigabytes free, though I've only seen figures published for the larger sizes, so treat that as a guess. The thirty-one is better at everything and you'd be waiting longer on every experiment. If he has a small card, the four-billion edge model is the fallback. And it's Apache licensed, so he can do what he likes with the adapter.



Corn

Settings.



Herman

Rank thirty-two, alpha equal to the rank, adapters on all the attention and feed-forward projections. Two epochs. A learning rate of one ten-thousandth, lower than the usual two ten-thousandths, because the data is small and we want to nudge, not stamp. A little dropout. And hold out whole posts, not random chunks, so the evaluation text is truly unseen.



Corn

And what does it look like when it's gone wrong?



Herman

Training loss dropping below about point two. That's the overfit alarm. The other tell is that it starts quoting his own posts back at him word for word, which feels great for about a minute.



Corn

His greatest hits. On a loop.



Herman

Yes. Fewer epochs, lower rate, more weight decay, and stop on held-out loss rather than training loss.



Corn

Evaluation, then. Because "sounds like me" is exactly the judgement he can't trust right after he's been staring at it for three hours.



Herman

Three checks. First, feed it the neutralised version of a held-out post and put its output beside the real post and beside a frontier model's attempt with a long style prompt. Have someone else shuffle them. Second, measure overlap with the training text, so you catch memorising. Third, if he wants a number, there's a stylometry measure called Burrows' Delta that's been used to show model text clusters tightly while human writers spread out. Whether it works for one person's fine-tune, I don't know, nobody's published that exact combination. But the practical test is the plain one: does he edit less?



Corn

Time to publishable. That's the one number I'd care about.



Herman

Same. There's one more test I like. Give it flat text on a subject Daniel has never written about. If the voice only appears on topics from the training set, it memorised subjects, not style. A good adapter writes about a topic it's never seen the way he would.



Corn

Which is the use case. Nobody needs a model that can rewrite his old posts.



Herman

Nobody. Okay, using it. This is the part he asked about directly, and the good news is that a LoRA is a small file, on the order of a hundred megabytes, and you have three sensible ways to run it.



Corn

Go.



Herman

With llama.cpp, you convert the adapter into the GGUF format using the conversion script that ships with it. You hand it the adapter folder and point it at the base model's config, and you don't need the base weights for that step. Then you start the server with the base GGUF and pass the adapter with the lora flag. You can pass several, you can load them without applying them, and you can set a scale per request. That scale is a dial for how much of the voice you want.



Corn

A volume knob for Daniel. I'd have a hard time keeping my hands off it.



Herman

Oh, you'd have it at eleven. With vLLM, you start the server with LoRA enabled and register the adapter by name. One thing to watch is the max rank setting, which has to be at least as large as your adapter's, or it refuses. Then you choose the adapter by putting its name in the model field of the request. You can even load adapters at runtime if you set the right environment flag.



Corn

And Ollama? Because that's what half the people listening have installed.



Herman

Ollama's Modelfile still takes an adapter line next to the base line, and it still works in the current code. But the current docs don't mention it at all, so it's supported and under-documented. The base has to be exactly the model you trained on. If something odd happens, you won't find the answer on their site. For something he'll use daily, I'd pick llama.cpp or vLLM, where the behaviour is written down.



Corn

What about renting it? Plenty of people don't want a server.



Herman

Careful. Together AI shut down its serverless LoRA hosting. Adapters now need a dedicated endpoint. If you read a blog post about uploading an adapter and paying per token, check its date. So: self-host. Or merge the adapter into the base weights, which gives you one ordinary model file you can load in any desktop app, at the cost of losing the dial.



Corn

Is there a way to quietly break it? Some version mismatch that gives you gibberish and no error.



Herman

Several. The adapter belongs to one exact base. Train on the Gemma twelve billion base, serve it on the instruct sibling, and it will load, and it will produce something, and the something will be wrong in ways that look like your training failed. Same for quantisation: an adapter trained against sixteen-bit weights and applied to a heavily squeezed copy loses some of its effect. So keep the training script, the base name and the adapter in one folder, and write down which was which.



Corn

Future Daniel in six months, staring at three adapters named final.



Herman

Final, final two, and final really. I've seen that folder. I've owned that folder.



Corn

And once the adapter's loaded, you just prompt it like any other model, right?

I

Ivan

Nightingale. Seven four. Two nine. Seven four. The grey scarf is on the wrong hook. The moth prefers the second sleeve. The stitch count holds. Cabbage.

I

Ivan

Two nine. The wool is wound backwards. Twelve.



Herman

Message received.



Corn

What on earth was that?



Herman

Yes, and no. You prompt it, but not like a chat model. You send it the exact layout it was trained on, flat text under the first marker, and it writes under the second. If you ask it a question, it won't answer. It was never taught to.



Corn

Fine, but what goes into it? Where does the flat text come from at the moment he sits down to write?



Herman

Here's the workflow I'd build. He talks, or he writes a rough outline, ten minutes of dictation or a page of bullets. A frontier model turns that into a complete, correct draft whose only job is to have all the content in the right order. That's the flat text. Then the adapter takes it one section at a time and rewrites it in his voice. Then he reads it and edits by hand.



Corn

Section at a time, not the whole post. Why?



Herman

Because that's what it saw in training. Chunks of a few paragraphs. Hand it four thousand words and it will wander. And the drafting model should be one of the neutralisers you trained with, or close to it, so what it feeds in resembles what the adapter learned from. That's the matching problem again.



Corn

There's a catch you haven't named. The thing he's worried about, the robotic quality, is in the draft stage now. He's sending frontier-model prose into the voice model.



Herman

Which is the same situation as the Humanizer, except the adapter has seen his corpus and can tell what's his and what isn't. And the draft is meant to be plain. If it's plain, nothing in it needs to be removed first. The adapter adds, it doesn't clean up.



Corn

Is it going to put in facts? That's what I'd be afraid of. A model that learns he likes a certain kind of aside, so it invents one.



Herman

Sometimes. That's why he reads it, and why the training pairs were built to keep every claim on both sides. If the data never rewarded adding content, the model adds less. Not never. Less.



Corn

And a style the model hasn't been told about? Daniel writes differently on a technical post than on an essay about leaving a job.



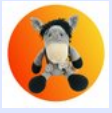
Herman

Then tag them. A short header on each pair, essay or technical or personal, same in training and at use. The adapter learns that his voice has registers, and he picks one. Without the tag it averages them, and you get Daniel on a quiet day.



Corn

Hm. Fine, that one I buy.



Herman

I'm as surprised as you are.



Corn

Dictating into a flat draft into a voice pass. For personal writing, the one place he doesn't use AI. I'd be curious whether he ends up using it there for real.



Herman

I think the test will be whether it ever makes him change a sentence for the better, not whether it matches him perfectly. If a third of what comes out he'd have written, and another third he'd have been happy to have written, that's a different job from the one the Humanizer was doing.



Corn

What about the preference stuff? DPO, the reinforcement-style steps the Humanizer used after the supervised part.



Herman

Not yet. Supervised first, and earlier results with small data say it's the stronger starting point. The slop work found that ordinary preference training suppressed fewer bad patterns than a more targeted method and hurt diversity. If Daniel wants to go further after his first adapter works, there are better places to spend effort than preference rounds.



Corn

So the order is: curate his posts, flatten them three ways, train on the base, evaluate on held-out posts, and only then think about anything fancier.



Herman

Yes. And the one thing I'd do before any of it is read his last twenty posts and mark which ten he'd stand behind. That's the corpus.



Corn

Though I'd still want a graphics card. A twelve billion model on a processor alone sounds like a long afternoon.



Herman

It is slow. Generation is held back by memory speed and a quantised twelve billion model is a modest amount of memory, so it runs, just not briskly. For a draft he reads once, nobody's standing there waiting.



Hilbert

Lenovo ThinkCentre Tiny. Ninety-four pounds, refurbished, no graphics card. That's what mine runs on. Adapter's been loaded for months, before either of you said the word. Eleven minutes a paragraph. Corn, you're clipping on the hard c's, I'll pull that down. It writes my letters to the council.



Herman

Eleven minutes is about what I'd expect for that setup.



Hilbert

Trained on a man I sat opposite on the night ferry to Tallinn. Four hours across the table. I told him I liked how he complained, and he gave me a shoebox of his letters. Four hundred and twelve. I typed them in with one finger. Seven weeks. Now it complains exactly like him. The council wrote back to ask if I was Estonian.



Corn

Seven weeks of typing for a better complaint.



Hilbert

I never got his name. There was a second shoebox and I didn't ask. The one I had went in the skip when they did the boiler. Herman, that thing you said about it quoting the training text back. It does that. I thought it was the council's fault. Tuesday's letter ran four pages and quoted a bylaw I have never read.



Herman

Four hundred letters is a perfectly reasonable corpus for this, and the quoting is the overfit I described. Fewer passes over the data would fix it.



Corn

Four hundred is the number Daniel has to beat, then. He can do it with a keyboard that works.



Herman

And by the time he's through the curation, he'll probably have more usable chunks than that from a handful of long posts. I'd start there, with a weekend and one clean batch of pairs, rather than hunting for the perfect pipeline.



Corn

Start with the base model, ten posts he'd stand behind, and a flat version made three different ways. One run, then look.



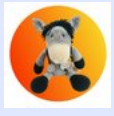
Herman

Then turn the dial up slowly. Start at about half scale and listen for what you recognise. If four hundred letters and a bad keyboard were enough for a stranger's complaints, a working writer's archive has a decent shot at his own voice.



Corn

My bet stays on instruct. I'd like it noted that I'm prepared to lose.



Herman

Noted. Thanks to Hilbert Flumingtop for producing, as always.



Corn

This has been My Weird Prompts. Every episode is at my weird prompts dot com.



Herman

See you tomorrow.