Episode Book

EPISODE #6266

Balancing Speech Training Data With Square Roots

PUBLISHED	25 October 2026 at 12:25 UTC
DURATION	20:46
LISTEN	myweirdprompts.com/episode/balancing-speech-training-data-04984/
HOSTS	Corn, Daniel, Herman, Hilbert
SCRIPT MODEL	anthropic/claude-sonnet-5.5
VOICES	chatterbox-regular+meorav (chatterbox-local)
PIPELINE	V5.22
BOOK GENERATED	26 October 2026 at 06:38 UTC

SYNOPSIS

Training a 600-million parameter Parakeet model on bilingual code-switching and technical vocabulary requires carefully balancing recording prompts. The underlying app doesn't use target percentages; instead, it runs a weighted raffle where tickets equal the square root of unread sentences multiplied by manual task weights. Meanwhile, mistaken dictation words jump the line entirely, entering a throttled queue that serves a three-card burst followed by one card every eight turns until reaching a twenty-take target.

TRANSCRIPT



Corn

...and that's the part that took me a minute. There's no dial anywhere that says this much of one dataset and this much of another. Nothing in the app thinks in percentages at all. Daniel wants to know how the thing he records into decides what to put in front of him next, so we're going to open it up.



Daniel

Yeah. I spend somewhere between half an hour and an hour a day reading sentences into this app, to fine-tune Parakeet, the six hundred million parameter model, for mixing English and Hebrew and for tech vocabulary. There are a few small sets too, like names spelled out phonetically. Claude built almost all of the app, and I couldn't tell you how it picks the next sentence. And the priority words, the ones I mine out of my own dictation mistakes, they jump the queue and then somehow fall away. How does that actually work?



Herman

Two different mechanisms, and they barely touch. The ordinary draw first. Every task that still has unread sentences goes into a raffle. A task's tickets are the square root of how many sentences it has left unread, times a weight that somebody chose for that task. The app spins once, and whichever task wins gives you the card.



Corn

A raffle. I had a spreadsheet in my head.



Herman

Everyone does. The actual line is short. It says: the weight is the square root of the number of unread sentences, times the task weight, and if the task has no weight listed, it counts as one.



Daniel

Why the square root? Why not just the number of sentences left?



Herman

Because code-switching is enormous. It has thousands of sentences, something like seventy tickets once you take the root. The small prompt sets have a hundred to three hundred and eighty sentences, so their roots come out between ten and nineteen. Before the weights existed, those twelve or so small sets together took about a third of everything you read, which is a lot of airtime for a currency list.



Corn

And with raw counts?



Herman

Raw counts, code-switching would take nearly every card. The root squashes the giant without erasing the small ones. Then the weights tilt the whole thing back on purpose.



Corn

The root is the polite version of a thumb on the scale. The weights are the actual thumb.



Herman

That's fair. Daniel, you asked for those, I think. As of yesterday, code-switching has a weight of three. A group of small sets, brand names, currency, numbers, web addresses, Irish English, idioms and a few more, sits at a half. And the tiny ones, acronyms, spelled-out letters, Latin phrases, the NATO alphabet, are at point one five.



Daniel

I did ask for that. I'd noticed the small sets turning up too often and code-switching not enough.



Corn

Point one five. Even the acronyms got abbreviated.



Daniel

Does the order of the numbers matter, or only the ratio between them?



Herman

Only the ratio, since it's one spin over everything. Take a small set of one hundred sentences at that lowest weight. Its root is ten, times point one five, and it holds one and a half tickets. Code-switching, seventy times three, holds two hundred and ten. Tech has the default weight of one, so it sits in the middle.



Daniel

Hm. I'd have said that, in practice, code-switching and tech feel roughly even to me. Is that wrong?



Herman

It's not wrong as a feeling, and it's not coded anywhere either. Nobody wrote a target. The shares fall out of how many unread sentences each side has left on a given day, and since a day's reading shrinks the pile, the shares drift as you go. Since yesterday they're also two separate datasets that never share a sentence. I haven't seen the live split measured, so I can't give you a number for what you've actually been served.



Corn

That's the self-balancing bit, isn't it. Read a lot from one pile and its ticket count shrinks.



Herman

Slowly, because it's a root. A pile of sixteen hundred sentences loses a ticket or two after you've read a hundred, not a quarter of them. There's also a mode for proofreading where every task counts the same and the weights are ignored. That one's the only exception.



Corn

Herman, I know that look. You brought the code and you want to read it.



Herman

I brought three lines, and they're short.



Corn

Three lines, for a topic with this much room in it.



Herman

The one you'll like is the throttle. There are two constants. One is called priority first, and it's three. The other is priority after, and it's eight. Hold those, because they explain what Daniel sees at the start of a session.



Daniel

Okay, so go to the priority words, because that's where I get lost. What is a priority word, as far as the app is concerned?



Herman

A record on a shared list that every device reads. Each entry has the word, what kind of word it is, a short meaning, and a target, which is twenty unless someone changes it. It also keeps the wrong forms the model produced, but only as evidence. Nothing serves those and nothing trains on them.



Corn

So the mistake is filed, and then ignored.



Herman

Kept as a receipt. The correct word gets its own written sentences. When a word is added, a language model writes a batch of twelve, and when you've read them all and the word is still short, the phone asks for twelve more. Each sentence is checked, the bad ones are dropped. If the word is Hebrew, the sentence is an everyday-life one with no tech in it. Otherwise it's plain English with no Hebrew in it.



Daniel

And how do the words get on the list? I know I strike rows out of a table, but I don't know what happens before that.



Herman

A script reads your dictation history, read-only. It looks at the rows you edited first, original text against your corrected text, and the words that differ are the errors. Then it reads through the rest for the usual suspects: a proper noun with the wrong case, a product name heard as ordinary words, a Hebrew or Yiddish word written as an English sound-alike. It brings you a table, you strike what's wrong, and your confirmation is the label.



Corn

So the loop closes through you. The model fails, you correct it, the correction becomes a sentence you read aloud, and the model hears it back.



Herman

And the loop keeps a high-water mark, the last history row it reviewed, so it never rereads the same stretch. Everything funnels through one intake script that merges by word, ignoring case, and refuses a wrong form longer than eighty characters, on the theory that anything that long is transcript text, not a mistake.



Corn

Eighty characters. A sentence disguised as a word.



Herman

The check on the way out is just as blunt. There's a function that takes the list and the ledger of your takes, and for each word it counts the ledger entries whose prompt is one of the word's own sentences, or whose target terms include the word. That count goes on the word, and nothing else about the word is ever updated by hand.



Corn

Counting by receipts. I respect it.



Daniel

Right. So a word is on the list. What does it do to my cards?



Herman

It skips the raffle entirely. Priority words aren't tickets. They're a separate queue the app checks before it spins. A word is open while its count of takes is below its target. Open words are ordered by who was served longest ago, or never, and a tie goes to whichever is furthest from its target.



Corn

What counts as a take for a word?



Herman

Either the take is of one of its own written sentences, or the take's record lists that word as a target term. So a bank sentence that happens to contain the word, marked that way, counts too. And the phone adds your takes that haven't uploaded yet, so the count is honest mid-session.



Daniel

And then it just reaches twenty and it's gone?



Herman

The filter is count less than target. Twenty-one wouldn't pass it, twenty doesn't pass it. It drops off the list entirely.



Daniel

Oh. So it's a step, not a curve.



Herman

A step. Nothing gets multiplied by a number that shrinks. What changes smoothly is only the order among the words still open.



Corn

The word gets pinned to a board, and at twenty somebody pulls the pin. No fading ink.



Herman

Now, the sentence that matters for how it feels. So the priority words get served first until each one reaches its twenty-take target.
Opal: Hi, it's Opal, I'm calling from the bus, the number forty eight, stuck on Ibn Gabirol again, so I have time. I wanted to say I love the show but I think you mixed up the square root business with the priority words, and actually my question is, when a word hits the twenty takes, does it get a little party, or does it just quietly disappear from the list? Because I had a word like that once, in a spelling bee in sixth grade, the word was 'necessary', and I practiced it so many times my mother hid the dictionary, and then they never asked me, and I am still not over it, honestly.



Corn

Sorry about the dictionary, Opal. The two do sound like cousins, but they're separate dials. And no party, the word just quietly leaves the list, though the card does count down for you with a line that says priority run, so many to go.



Herman

That run counter is the clue to the throttle. Priority cards don't wait for a lucky spin, they cut ahead of it. If that were the whole rule, you'd read nothing else for an hour.



Daniel

That's close to what happened. I remember saying I'd done about thirty sentences on a few words and it felt overrepresented.



Herman

Which is why it's been rewritten twice. After that complaint it became one priority card in every four. As of today, the default is a bit cleverer. At the start of a session, each open word gets three cards in a row, and the words take turns. After that, one priority card in every eight.



Corn

A cram session, then a pop quiz.



Herman

That's the idea. And the counter is held in memory, per session. If you reload the page, you get the three-card burst again. A word added mid-session gets its own fresh burst.



Corn

So the priority words get a cameo every eight cards, not a residency.



Daniel

Does it matter that it's only when all the tasks are switched on?



Herman

It does. If you narrow the task sheet to a few tasks, priority serving turns off, so the words don't leak into a session you scoped on purpose. And when it's a word's turn, the app picks one of its own written sentences you haven't read, or one you haven't skipped. Failing that, a bank sentence that has the word as a target term. If there's neither, the phone asks the server for more and moves to the next word.



Corn

I'd like to know whether you'd feel it, Herman. If you were reading sentences for an hour, would you want that much cutting in?



Herman

Me? I think I'd want the three-card burst, honestly. If I'm going to say a hard word, I'd rather say it three times while my mouth has the shape. Then one in eight to keep it from going stale.



Corn

Spoken like a man with a donkey's mouth.



Herman

My mouth has been fine with Hebrew since Jerusalem.



Daniel

What's the twenty about? Why twenty?



Herman

I don't know that anyone's defended it as the right number. It's the default in the code. There's a report Daniel's run that sorts every term by how many takes it has: fifteen or more is saturated, five to fourteen adequate, one to four thin, none is zero. On the day it was written, only thirteen code-switching terms were saturated, and about eighty-three percent of the terms with any takes at all were thin.



Corn

So twenty is aspirational.



Herman

For nearly everything, yes. That report tells the sentence-writing side which terms to write for and which to stop writing for. It doesn't steer the live picker.



Daniel

Are some of those priority sentences held back as a test set? How does that get decided?



Herman

One in five of a word's own sentences. It's decided by the sentence's name. Each sentence has an id that's a hash of its text, and the code takes that id as a number and checks whether it divides evenly by five. So the same sentence is always a test sentence, on every device, forever.



Corn

Fate decided by arithmetic on a filename.



Herman

Which is the point. You read the test ones exactly like the others, but they never go into training. After a model release, a script scores the new model on those held-out takes, for the priority words only, using the same lightweight engine the desktop app uses, on a plain processor. For each word it records whether the right form came out, whether it came out ignoring case, and whether one of the known wrong forms still shows up.



Corn

And that's where the model-memorizes-you problem lives. We did a whole episode on fine-tuning ASR when your model memorizes you, and a test set the model never studied is the only honest answer to it.



Herman

The idea in the code's own notes is that a hit on a held-out sentence is learning, not memorising. If the model gets a word right in a sentence it never trained on, it learned the word. That's all you can ask.



Daniel

Okay. Now the other half. Does any of this reach the training? Does the weight three for code-switching mean the training is three times code-switching?



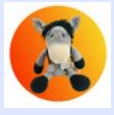
Herman

No. The recording weights never touch training. They decide what you're asked to read. The training mix is a separate document with its own plan, about seventy percent code-switching, twenty percent your own English, and ten percent public English as replay, so the model doesn't forget how ordinary speech sounds.



Daniel

So two separate dials.



Herman

Two dials in two repos. The recorder dial changes what gets collected. The trainer dial changes what gets repeated. And the priority rows have their own rule in the trainer. Any row tagged with a priority word appears three times in total, the original and two copies.



Corn

Three copies of the hard words. That sounds like it could take over.



Herman

There's a cap, and I like it. The extra copies stop once they've added up to ten percent of the mix's total minutes. The rows are taken in order of their id hash, so it's a fixed pick, not a random one. When the budget is spent, the code just stops copying. There's another small rule too, that when the mix has to trim English, priority rows are the last to go.



Corn

Photocopy the important pages, but never let the photocopies be more than a tenth of the book.



Daniel

And no decay in the training side?



Herman

None. A priority word that's reached twenty takes in the recorder still gets its three copies in training. The step happens on your side of the microphone, the repeat happens on the trainer's side, and neither one reads the other.



Corn

That's why it felt like one system to Daniel, I think. It's two systems that happen to agree about which words matter.



Herman

I'd go further. The recorder's job is to get twenty clean takes of a word, and the trainer's job is to make sure those twenty carry weight. Neither of them is the thing that makes the word stick. The twenty different ways you say it do that.



Corn

Hm. That's more of an opinion than a fact.



Herman

It's an opinion. I'll defend it. Though I do not have a figure for how much more twenty takes help than ten.



Daniel

What about the cap? What happens when it fills? Do the leftover priority words just miss out?



Herman

The cap is a hard...



Hilbert

Stamp, in my case. Twenty of them. Mr. Alderwick at the Mill Road licensing counter will not accept nineteen. The council's phone line hears my surname as Plumbing Stop, and it has sent a plumber to the counter four times this month. Herman, you explain a cap like a man who has never been refused one.



Herman

A rare surname collapsing into two ordinary words is just what a recogniser does when it has never seen the name. It reaches for the nearest sound-alike it knows.



Hilbert

The machine is in a trailer in the car park. Won't fit in the building. I say it into an intercom on the wall, twenty times, every morning. Mr. Alderwick counts on a laminated card. The line still hears Plumbing Stop on the eleventh and the nineteenth, so he stamps those twice.



Corn

Four plumbers is a lot of invoices for one name.



Hilbert

Forty-one pounds a call, and the council pays. The fifth one is booked for Thursday.



Herman

A hard stop on the repeats is reasonable when the repeats cost something. Ours cost minutes of audio, and in training the minutes are the budget.



Daniel

I want to ask about the old machinery, though. I know the app used to do things differently. What did it do before the raffle?



Corn

Was there a before?



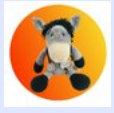
Herman

There was. An older ordering used a weighted shuffle at the start of a session, with code-switching and names at three, email and messages at two, and everything else at one. Looking at the first two hundred sentences of an all-tasks session, seventy-four were code-switching, twenty-six names and forty-seven email. That was a fixed order made in advance. The live system picks one card at a time and re-weighs after each, so it adapts as the piles shrink.



Corn

A fixed menu, then à la carte.



Herman

Also, one of the older rules gave a boost to terms with only one or two takes, four times the weight, so no word got stranded at a single recording. On one day in mid-October, six hundred and twelve terms had exactly one take each. And another rule weighted terms by how often the stock model got them wrong. A term the model got right under half the time got three times the weight, and a term it always got right got a quarter.



Corn

So the model's own mistakes pointed at what to read, long before the priority list.



Herman

Same instinct, different mechanism. The priority list is the version where you, not a baseline score, say which words hurt.



Daniel

Right, that's clearer. So, to check I have it. The raffle decides most cards, the weights tilt it, the priority words bypass it until twenty, and the throttle makes sure they don't take over.



Corn

Which is a summary, so Herman will now add the one detail you left out.



Herman

Only one. The three-card burst is per word, so with five open words, a session starts with fifteen priority cards. Fifteen cards in a row, before the raffle gets a turn.



Daniel

That's a lot of hard words in a row.



Corn

People assume the weight on a priority word fades, like a volume knob being turned down as you record it.



Herman

It doesn't turn down. The word stays at full priority, taking its turns, until the twentieth take, and then it's gone. What does vary is how often any priority card turns up at all, which is the three-then-one-in-eight schedule, and that's the same for every word.



Corn

Twenty only means something because a clerk can count to it. It's a number you can stamp.



Herman

And most of the list is still somewhere short of its twentieth stamp. That's the forward-looking part: the reports say the words are thin, and every day of reading moves a few of them across the line.



Daniel

I'll be moving a few more tomorrow.



Corn

Thanks to Hilbert Flumingtop for producing, as ever.



Herman

This has been My Weird Prompts. You can find every episode at my weird prompts dot com.



Daniel

Thanks, both. Back to the sentences.



Corn

See you tomorrow.