Episode Book

EPISODE #6508

Why Speech Transducers Omit Common Symbols

PUBLISHED	9 November 2026 at 13:50 UTC
DURATION	25:15
LISTEN	myweirdprompts.com/episode/speech-transducer-vocabulary-limits-28586/
HOSTS	Corn, Herman, Larry, Hilbert
SCRIPT MODEL	z-ai/glm-5.3
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.26
BOOK GENERATED	10 November 2026 at 10:05 UTC

SYNOPSIS

When speech models fail to produce common characters like email at-signs, the limitation stems from deep architectural tradeoffs rather than missing training data. RNN-T and TDT architectures calculate a four-dimensional joint tensor that scales directly with vocabulary size, making large vocabularies memory-prohibitive. This architectural constraint separates models into two camps: massive LLM-style decoders like Whisper that emit symbols directly, and compact transducers like Parakeet that rely on external inverse text normalization layers to format raw spoken words downstream.

TRANSCRIPT



Corn

Daniel walked face first into a wall this week, and the wall turned out to be about four pixels wide. He was adding training sentences to his fine-tuned Parakeet dictation model so it would recognise email addresses, and then it hit him: Parakeet's output vocabulary has no at-sign. Not "rarely emits" — there is no token for it. You could feed it a million sentences with email addresses and it physically cannot write one.



Herman

And that stopped him, because he knows enough about the architecture to know it's not a bug he can train around. But then he looked sideways at the rest of the field and noticed the models split into two camps, and his instinct — his word, "instinct" — is to be suspicious of NVIDIA's side of the split. Why keep the speech model's vocabulary small and bolt on all these little formatting layers after it, when Qwen3-ASR and Whisper just write the symbols out themselves? How much can three extra characters really cost?



Corn

Which is exactly the right question to hinge it on, because the answer is not what you'd guess. And I want to say upfront, this is one of those stories where every layer you peel off has another layer under it. Daniel thought he'd found a bug. He found a philosophy. Let's start with why the wall exists at all.



Herman

The wall is architectural. Parakeet-TDT-0.6b-v2 has an output vocabulary of roughly one thousand and twenty-four tokens, built from its training transcripts. Somebody did a CoreML export of the v2 vocabulary, and you can read the whole symbol set right there: period, comma, hyphen, apostrophe, question mark, colon, exclamation, percent, dollar, slash, caret, pound, euro, the upside-down Spanish ones. Slash is in. Dollar is in. The at-sign is not.



Corn

So it's not that the model is illiterate. It punctuates, it capitalises, it'll happily write a slash for you. It just never learned the one character that lives inside every email address on earth. And I want to be precise about what that means in practice, because it's stranger than "it makes a typo". If Daniel dictates "herman at example dot com", what comes out the other end?



Herman

Whatever the training transcripts taught it to say when someone speaks an address. Most likely "at" as a word, or some subword approximation. The model isn't failing to write the symbol. It's succeeding at writing the only thing it can write. There's no error message, no confidence drop. It just confidently produces the wrong thing, forever, in every context.



Corn

Right, and the reason it can't be trained in is the part I think people don't picture. The model can only emit what sits in its output dimension. When you give it a training sentence containing an at-sign, there is no target token for that character to map to. The training loop has nowhere to put it. It's like... you know how a hotel keyboard sometimes has no key for a letter in your language? Pressing harder doesn't help. The training data isn't the constraint — the socket is. And this is worth pausing on, because the fine-tuning instinct is always "more data". You hit a ceiling, you go get more sentences. Here, more sentences is a category error. More sentences containing the at-sign just teach the model to skip over it or swap in something it does know. The symbol never enters the building.



Herman

So Daniel's first question — is this common across ASR models — the answer is: it's common across transducer-style models with small output vocabularies, and it's basically unheard of in the LLM-decoder camp. Whisper has a byte-level vocabulary of about fifty-one thousand tokens, so every byte of every symbol is representable. Qwen3-ASR decodes through a Qwen language model, so it inherits an LLM-sized vocabulary, well over a hundred thousand tokens. Parakeet's whole output world is a thousand tokens. That's the split, and everything else today hangs off it.



Corn

So there are two opposite philosophies about where written formatting belongs, and I think the vocabulary question is where it gets interesting. One camp says inside the model: the decoder writes symbols, digits, punctuation, done, one artefact, simple. The other camp says outside it: the speech model stays small and speaks plainly, and a separate inverse text normalisation layer downstream turns spoken form into written form. And I'll admit, when you phrase it that baldly, camp one sounds obviously better. One artefact, fewer moving parts. Why would anyone choose the second?



Herman

Hold that thought, because the answer is a number we'll get to later, and it's a big one. But first — Larry has something he'd like to interject.

L

Larry

Friends, have you ever tried to type an at symbol into a speech model that simply does not have one? Of course you have. That is why I am thrilled to tell you about the Vocab Vault, the only limited-time offer where the price drops every single second I keep talking, and I have no intention of stopping. We started at two thousand and one dollars, we are already down to nine hundred, and hold on, now four hundred and twelve, and I will throw in a second vocabulary, a spare inverse layer, a small laminated grammar, and a bag of unrelated extras you never asked for, including a decorative semicolon and a genuine imitation pilcrow.

L

Larry

Three hundred, two hundred, one hundred and ten dollars, and the deal ends before I finish describing it, so act now, or honestly, do not, because it is already over. Smooth.



Herman

Hang on, a genuine imitation pilcrow... Corn, a pilcrow is already just a mark on a page, how do you imitate a...



Corn

Herman. Let it go. The price is already over.



Herman

...Right. Where were we. The two camps. So the interesting wrinkle is that NVIDIA is actually in both camps at once. Canary-Qwen, the two-and-a-half billion model they put out in July of last year, pairs a FastConformer encoder with a Qwen decoder — a real LLM decoder — and outputs punctuated text directly. So inside one company you have the thousand-token transducer with the separate ITN layer, and the LLM-decoder model that writes formatting itself. They're hedging.



Corn

Which tells you neither philosophy has won. Let's do the mechanism first, because I don't think the skimpy question can be answered without understanding why a thousand tokens and not fifty thousand.



Herman

It's memory. The RNN-T family — and TDT, which is Parakeet's variant — computes a joint tensor that is batch times audio frames times output tokens times vocabulary. Four dimensions, all multiplied together, all held at once. The Pruned RNN-T paper from Interspeech 2022 says flatly that this loss "can use a lot of memory" and becomes impractical with a large vocabulary — they cite character-based Chinese as the case that breaks it.



Corn

Run the arithmetic for me.



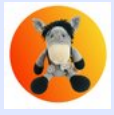
Herman

Rough numbers, but they're honest rough numbers. Take a batch of thirty-two, two hundred and fifty encoder frames — that's about twenty seconds of audio — a hundred output tokens, fp32. At a vocabulary of about a thousand, the joint tensor is around three point three gigabytes. Fine. At a Qwen-sized vocabulary of a hundred and fifty-one thousand, the same tensor is about four hundred and eighty gigabytes. Before gradients.



Corn

Four hundred and eighty gigabytes for one batch of twenty-second clips. Let me make that concrete for people, because gigabytes float past. An H100 has eighty gigabytes of memory. So this isn't "you need a bigger GPU" — this is "the tensor is six times the memory of the whole card, and that's before you've stored a single weight."



Herman

Before gradients, before the optimizer state, before anything else you wanted to keep in that GPU. That's why the vocabulary is small. It's not an aesthetic choice. Every token you add multiplies through frames and output steps. The at-sign is one more column in a tensor that already eats three gigs at a thousand tokens.



Corn

So the question flips. It's not "why is the at-sign missing", it's "why are slash and dollar present". Somebody chose those ten-ish symbols deliberately and left the rest to the layer downstream.



Herman

That's a fair reading. And the LLM-decoder camp dodges this entirely — an autoregressive attention decoder's cost grows with tokens times vocabulary. There's no frames-times-tokens-times-vocab tensor anywhere. That's the structural reason Whisper can afford byte-level coverage: the memory profile of an attention decoder doesn't punish vocabulary the way a transducer does.



Corn

Okay, so suppose Daniel bites the bullet and adds the token anyway. New tokenizer, one thousand and twenty-five tokens. What actually happens?



Herman

The output dimension changes, and NeMo's own fine-tuning docs are explicit about what that costs you. For a transducer, when the tokenizer changes, the decoder — the prediction network — and the joint network both get re-initialised. Only the encoder survives. You are no longer fine-tuning. You're retraining two of the three components of the model from random weights, using the encoder as a warm start.



Corn

So his careful fine-tune, the one that already knows his dictation style...



Herman

The encoder part of it survives. The part that actually emits tokens starts over. And the docs tell you to use learning rates around ten to the minus four or lower, ten to the minus five, because if you're too aggressive you destroy the pretrained features in the layers you did keep. There's also a community report from somebody who swapped a Parakeet TDT tokenizer — once they unfroze the encoder, training collapsed into the model just repeating common subwords over and over. That author's conclusion was don't attempt a vocabulary swap with under a thousand hours of data. One person's experience, but it's consistent with how fragile the joint is after re-initialisation.



Corn

A thousand hours. A thousand hours of audio is roughly a person speaking nonstop, day and night, for six weeks.



Herman

To be fair, that warning was about swapping the whole vocabulary. Bolting one token onto the existing one is gentler, but it still means the half of the model that emits text gets retrained, so you're nowhere near a quick fine-tune. There's a second problem too, and honestly it might be the deeper one. Even if the socket existed, the data doesn't. Most public ASR corpora are transcribed in spoken form — LibriSpeech is uppercase and unpunctuated, all of it. Parakeet v3 uses a unified eight-thousand-token SentencePiece vocabulary across twenty-five European languages, but if you read its card, the reported word error rates are computed after stripping punctuation and capitalisation from both the reference and the hypothesis.



Corn

Wait. The headline numbers literally don't measure the thing we're arguing about.



Herman

Not at all. Strip punctuation before scoring and the benchmark is blind to whether the model writes "at" or the symbol or nothing. Nobody's headline metric would move if every at-sign in the evaluation set vanished tomorrow.



Corn

That's the quiet part, isn't it. The benchmarks are scored in a way that makes the missing symbol invisible, the training data barely contains the symbol, and the architecture punishes adding it. Three walls, all pointing the same direction, and none of them is stinginess.



Herman

So to answer Daniel straight: yes, it's common across transducer-style models with small vocabularies, and no, no amount of training data fixes it — the fix is a tokenizer change that re-initialises the decoder and joint, which is a different project from fine-tuning. We asked a cousin of this a few weeks back, whether cleanup belongs inside the speech model or in a second model. This is the same fork, one level lower: not cleanup, but which characters the model can even produce.



Corn

Right — the mechanism's done. Now the interesting half: why NVIDIA chooses to keep formatting out of the model entirely, and whether the little layers are actually little.



Herman

So the thing downstream is called inverse text normalisation, and I want to defend it properly, because Daniel's picture of it — "little layers that turn at and dot com into symbols" — undersells it in a way that matters. NeMo's ITN is built from weighted finite-state transducers. Grammars, essentially — enormous, hand-built rule systems in Pynini and OpenFst that can even be exported for Google's Sparrowhawk runtime if you want them in production serving. They're organised by semiotic class, the Sproat and Jaitly taxonomy: cardinal, date, money, electronic, and so on. Each class is a classify grammar that spots the span, then a verbalize grammar that rewrites it.



Corn

And "electronic" is where emails live.



Herman

It is, and the worked example in the source is almost comically literal: "c d f one at a b c dot e d u" comes out as cdf1@abc.edu. "Dot" becomes a period, "colon slash slash" becomes the full protocol separator. The grammar knows the spoken choreography of an address.



Corn

Which immediately hands you the misfire. If "at" and "dot" are fixed keywords, then "I'll be at the station around five, dot, whatever comes next"...



Herman

The source admits it. The keywords can misfire outside addresses, and — this is the structural limit — ITN cannot recover from words the ASR got wrong. It operates on the transcript, whatever the transcript says. If the model heard "add" instead of "at", the grammar has nothing to work with.



Corn

Okay, so the layer has teeth-marks. What's the actual case for it? Because Daniel granted there must be one and I don't think it's charity.



Herman

Three reasons, and I find all three real. First, determinism. Because the rules are context-free transducers, "one slash four" always maps to the same output. It never gets creative, it never hallucinates a date because the surrounding sentence felt date-ish. The trade-off, which NVIDIA's own blog states plainly, is heavy build effort, combinatorial bursts in the grammars, and scarce labelled data. You're paying engineers instead of paying GPU hours.



Corn

Which is a very NVIDIA trade.



Herman

Second, it's per-language and swappable. Riva runs ASR, then a separate punctuation-and-capitalisation model, then ITN, and some Riva models ship with no ITN support at all — it's a per-language add-on. You can update the money grammar for a currency change without touching the acoustic model. Third, and this is the one I think actually drives everything: speed. On the Open ASR Leaderboard, Whisper large-v3 transcribes at about a hundred and forty-six times real time, and Parakeet v3 at about three thousand three hundred times.



Corn

Twenty times faster, with a better average error rate.



Herman

Better! Six point three two versus seven point four four, and more than twenty times the throughput. That's the whole game. The small-vocabulary transducer is the speed play, and the ITN layer is where all the formatting complexity gets parked so it never touches that tensor we talked about. To be clear, three extra tokens wouldn't slow Parakeet down at all. What would is going the whole way, to a fifty-thousand-token vocabulary or an LLM decoder. Once you're adding symbols one at a time, you have to decide where to stop, and NVIDIA's answer is: stop early, and let the grammar handle the rest.



Corn

So the answer to "why be so skimpy" isn't the characters themselves. Let me make sure I've got this. The marginal parameter cost of an at-sign is one output unit — nothing. The cost is retraining the decoder and joint from scratch, and needing transcripts that actually contain the symbol, which mostly don't exist.



Herman

That's it exactly. NVIDIA isn't counting pennies on characters. They're being conservative about what a small-vocabulary transducer can be retrained into without losing the thing that makes it worth deploying. The at-sign is cheap; the road to it is expensive.



Corn

Though I'd want the other camp's costs on the table too, because "the LLM decoder just writes the symbols" is doing a lot of load-bearing work in that sentence.



Herman

They're real costs. Whisper large-v3, given non-speech input — silence, breathing, ambient noise — produced text on forty percent of those segments in one ICASSP study. Forty percent. It invents. And about one percent of transcriptions in another study contained content that wasn't supported by the audio at all. The flexibility you need to write an at-sign is the same flexibility that writes a sentence nobody said.



Corn

The at-sign is not free. It's bundled with a lying module.



Herman

That's the trade in one line, yes. Qwen3-ASR's numbers are self-reported, so take them with the usual salt — LibriSpeech-other three point three eight against Whisper's three point nine seven — but they're in the same room. And the neural-ITN route has its own embarrassments: NVIDIA built a tagger that can duplicate words and a Duplex model that can hallucinate, and their own blog concluded that the WFST-plus-language-model combination was least affected by unrecoverable errors. There's also an ICASSP paper from a couple of years ago showing neural ITN trained on clean text stumbles badly on real ASR output, which is the only input it ever actually gets.



Corn

So both camps have a scar. The transducer camp can't write the symbol; the LLM camp writes symbols that shouldn't be there.



Herman

And the most interesting recent hit against the separate layer is Dual-Form ASR, a Chinese paper this year. Their argument is the error-propagation one we already touched: spoken-form ASR followed by ITN is fragile because the first stage's mistakes cascade into a stage that can't correct them. Their fix is a single model trained on both forms, spoken and written, and you pick the form by prompt. They report it beats the open-source ASR-plus-ITN pipelines.



Corn

One model, a switch. Which is... close in spirit to where Canary-Qwen is heading.



Herman

Which is why I said NVIDIA is hedging. They built both. And here's the bit that keeps this from resolving: nobody has run the head-to-head. I couldn't find a single benchmark of written-form end-to-end versus spoken-form-plus-ITN on email or URL accuracy. The entire argument is architectural. Everyone's reasoning from the shape of the pipelines, not from measured outcomes on the one task we care about. The whole argument is about where the at-sign should be born, and nobody has ever counted how often each design gets it right.



Corn

So anyone with a few thousand labelled email dictations could settle a question the two camps have been circling from opposite sides. That's rare in this field, a settled-by-next-Tuesday open question.



Herman

Daniel could settle a chunk of it himself, honestly — he's got the fine-tune, he's got the domain. Compare his Parakeet plus the electronic grammar against a small LLM-decoder model on his own dictation, count at-signs.



Corn

He'd need the electronic class to stop attacking "at the station", but yes. My money says the speed gap means the separate layer isn't going anywhere soon. Twenty times throughput buys a lot of forgiveness.



Hilbert

It's not one character, that's the thing.



Hilbert

The at-sign is two pieces. An a, and a tail wrapped around it. On the printer's block I had re-keyed in Belfast, the a and the tail were cast separately and assembled by hand, and I have never trusted the symbol since. I still have the two pieces. A small tin. I look in on them now and then.



Corn

You check on them.



Hilbert

They came apart once. That's all I'll say about that.



Herman

The tail separate from the a... actually, on the CoreML export we were discussing, the symbol would be a single token in the vocabulary — one index. The two-piece thing only exists in metal.



Hilbert

Herman, you spent ten minutes on that tensor and not one second on the real scandal, which is that the grammar treats "at" and "dot" as fixed keywords. Say "I'll be at the office" near a URL and it will build you an address out of nothing. That happened to me this month. I dictated a note to the council about the bins, told them I would be at the depot, and the reply came back addressed to a Mr I'll Be, at depot dot com. It took me three



Hilbert (continued)

letters to get a human. I type my addresses with these claws now, one digit at a time, and I've stopped complaining about it.



Corn

The claws are doing the job the grammar wouldn't.



Hilbert

They're doing it slowly, but they've never once invented a Mr I'll Be. That's the bar the machine has to clear.



Corn

It's a low bar, but it is a bar.



Herman

The keyword thing is fixable in principle — the classify grammar could demand a domain-ish context before the verbalize grammar fires — but every constraint like that is another path through the transducer, and the build effort is exactly what NVIDIA's blog complains about.



Hilbert

Then they can keep it. I'll keep the tin.



Corn

Fair division of labour. Before we wrap, one misconception I want to kill, because it's the one everyone arrives with: that more training data would have fixed Daniel's problem. It wouldn't. The tokenizer defines the output dimension — the at-sign had no token, so no sentence containing it could teach the model anything. The fix was never data; it was re-initialising the decoder and joint, which is a rebuild, not a tune.



Herman

And the twin of that one: that NVIDIA is being cheap about characters. One output unit costs nothing. A thousand hours of retraining and transcripts that barely contain the symbol — that's what they were declining to buy. The skimpiness is in the retraining bill, not the token budget.



Corn

The open question I'd leave with is Canary-Qwen. If NVIDIA's own model shows the LLM decoder works for them, then the separate ITN layer is either a legacy architecture on its way out, or a deliberate speed play that keeps latency-sensitive deployments — the twenty-times-throughput ones — on the transducer for years. Both readings fit everything we know.



Herman

Since nobody's benchmarked email accuracy either way, the next move belongs to whoever labels a few thousand addresses first.



Corn

Our producer is Hilbert Flumingtop, who has never once invented a Mr I'll Be. If you enjoyed this one, try episode seven, Building Custom ASR Tools. This has been My Weird Prompts. If the episode earned it, leave us a review wherever you listen.



Herman

Rate the at-sign on your way out. It's two pieces, apparently. We'll be back soon.



Corn

See you tomorrow.