Episode Book

EPISODE #5994

Hebrew in Latin Letters: A TTS Pipeline Fix

PUBLISHED	15 October 2026 at 13:30 UTC
DURATION	33:28
LISTEN	myweirdprompts.com/episode/hebrew-latin-token-classifier-46654/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.4
BOOK GENERATED	19 October 2026 at 17:42 UTC

SYNOPSIS

When your text-to-speech reads "makolet" as MACK-oh-let, you have a language identification gap, not a bug. No LID model — GlotLID, fastText, NLLB's lid218e — has a heb_Latn label, so Hebrew written in Latin letters is invisible to every system downstream. This episode walks through the token classifier Daniel shipped to fix it: BIO tagging over xlm-roberta-base, character-span output, and a three-pass pipeline that converts Latin-script Hebrew to Hebrew script before handing it to a Hebrew voice. The real story is the dataset methodology — term-seeded generation, blind annotation by a different model family enforced in code, human review of disagreements, and term-disjoint splits. Plus why precision beats recall here, why a flagged span is work rather than a label, and what a Parakeet fine-tune on dictated audio adds to the picture.

TRANSCRIPT



Corn

The show's own pipeline is the reason this thing exists. Every time an episode script has a Hebrew word in it, our text-to-speech reads it as English letters and murders it.



Herman

Which we have been hearing for weeks now.



Corn

Daniel sent us the release notes for a model he shipped today. Hebrew-in-Latin token classifier, public on Hugging Face. It finds Hebrew words written in Latin letters inside English text, so a TTS pipeline can route those spans to a Hebrew voice instead of mangling them. And then he wants to know why anyone would want it, what it does, how the dataset was built, what base model, how to wire it into a pipeline, and what the audio version of the same idea would be for. Plus a Parakeet fine-tune that landed the same day.



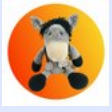
Herman

Bituach Leumi before the makolet closes.



Corn

That is the example he leads with, and it is the whole problem in one sentence. Makolet is not an English word. But it is written in English letters, so every system downstream treats it as English.



Herman

And that is not a bug in any one of them. It is a gap. GlotLID, the fastText lid model, NLLB's lid218e, none of them have a heb_Latn label. There is no category for this. The model cannot say "this is Hebrew written in Latin letters" because that label does not exist in the taxonomy.



Corn

So the text is invisible.



Herman

Invisible to language identification, yes. And there has never been a token-level code-switching model covering Hebrew at all. The LinCE benchmark has Spanish, Hindi and Nepali mixed with English. Hebrew is not in it.



Corn

There is something almost funny about that. The tooling has a slot for Hindi in Latin script but not Hebrew.



Herman

It is a bit funny. And the mirror direction was studied. Goldberg and Elhadad, back in 2008, looked at foreign words written in Hebrew script. So somebody studied Hebrew as the host language absorbing foreign words. Nobody studied Hebrew as the guest.



Corn

The reverse migration.



Herman

Exactly the reverse. Which means when Daniel says there was no model for this before, he is not exaggerating for effect. There was not.



Corn

So what does the model actually output?



Herman

BIO token classification over xlm-roberta-base. O, B-HE, I-HE. Multi-word terms come out as a single span. Bituach Leumi, kupat holim.



Corn

And the North Star word is makolet.



Herman

The project has a stated rule. If a rule would exclude makolet, the rule is wrong. That is the test everything else has to pass.



Corn

Let me guess why. Makolet is the ordinary corner grocery. It is not a religious term, it is not a holiday, it is not something an English speaker has ever had a reason to learn. It is just a shop.



Herman

And it is the exact category of word that falls through every gap. It is short, it is colloquial, it looks like it could be English, and an English voice reads it as MACK-oh-let.



Corn

MACK-oh-let.



Herman

That is what our pipeline does with it today. It stresses the first syllable, it rhymes it with something, it is plausibly a name of a person.



Corn

So the classifier's job is to find the span and hand it off.



Herman

Emit character spans. Not a verdict, not a rewrite. Just here are the characters, here is where they start, here is where they end. Then pass two converts those to Hebrew script. Pass three hands them to the TTS engine with a Hebrew language identifier.



Corn

Before we go into the pipeline, I want to push on the labelling rule, because that is where the interesting argument is. What counts as Hebrew-in-Latin?



Herman

The test is pronunciation, not etymology.



Corn

Say more.



Herman

The question is not "is this word Hebrew in origin." The question is "would an English voice mispronounce this, and would treating it as Hebrew fix that." So Shabbat, hummus, challah, kosher, aliyah, and city names by default are not flagged.



Corn

Deliberately.



Herman

Deliberately. English already says those acceptably. If you flag it, you add work.



Corn

And that is the phrase I want to keep. A flagged span is not a label, it is work.



Herman

That is the framing in the project. Because the asymmetry is not symmetric. A false positive means an English word gets read in Hebrew. That costs you a sentence split and an extra TTS call. A miss costs you nothing.



Corn

Nothing?



Herman

Nothing. If you miss makolet, you get what you have today, which is a slightly wrong pronunciation. If you false-positive on "salon" and route it to a Hebrew voice, you have actively broken an English word that was fine.



Corn

So the default posture is conservative because the costs are lopsided.



Herman

And they are lopsided in the direction that most people building classifiers would not guess. Everyone assumes recall is the virtue. Find more, catch more. Here, find less, catch less. Precision is the whole game.



Corn

Which is a nice inversion. Most of the time you are trying to not miss things. Here you are trying to not touch things.



Herman

Right. And the harder part is that the boundary is fuzzy. "Shakshuka" is on the miss list and it is a dish most English speakers have seen on a brunch menu. "Sukkot" is a holiday and it is also a common loanword in Israeli English. Those sit right on the line.



Corn

So how do you build a dataset where you can defend the line?



Herman

That is the methodology, and it is the intellectual core of the whole release. Four design rules.



Corn

Number one.



Herman

Term-seeded generation. The generator is given one specific term and asked for sentences using it. So the span is known by construction, not by searching for it afterwards.



Corn

That sounds obvious until you see what it prevents.



Herman

It prevents the generator from wandering. If you just say "write me sentences with Hebrew words in them," you get whatever the model feels like, and then you have to find the Hebrew after the fact, and finding it after the fact is exactly the failure mode we will come to.



Corn

Also, word-boundary rejection is built in. If the seed term does not word-boundary match in the generated sentence, the sentence is thrown out.



Herman

Thrown out. Which matters because of a predecessor dataset. The English-Hebrew mixed sentences corpus, built for the Whisper-Hebrish model, had labels derived from substring search. Twenty-four percent of those labels matched only inside other words.



Corn

Give me the examples.



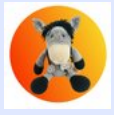
Herman

"Hi" inside "think." "Har" inside "Pharm." "Chool" inside "school."



Corn

Hold on. "Chool" from "school."



Herman

The whole tail of the word.



Corn

And these looked fine?



Herman

That is the point. A substring check reported zero failures on those labels. Zero. You run the check, it says the substring is present, everything passes. Only word-boundary matching exposes that the label is inside a different word.



Corn

So the previous dataset was passing its own tests while being wrong about a quarter of its labels.



Herman

Which is the lesson that drives rule one and rule two. Twenty-four percent is not a rounding error. That is a dataset that would train a model to flag the end of the word "school" as Hebrew.



Corn

Rule two.



Herman

Blind annotation by a different model family. The annotator never sees the seed term. And the build script refuses to run if the generator and the annotator share a model family.



Corn

Refuses to run.



Herman

Refuses. It throws an error. Because if you use the same model to generate and to annotate, and it agrees with itself, what have you measured?



Corn

Consistency.



Herman

Consistency. Not correctness. It is the difference between a model confirming its own opinion and a second party checking the work.



Corn

I like that it is enforced in code rather than in a policy document. People write lovely policies about independent annotation and then use the same vendor for both.



Herman

And there is no way to accidentally comply. If the families match, you do not get a warning, you get a crash.



Corn

Rule three.



Herman

Disagreements go to human review. And there is a mandatory ten percent random audit of the agreements. Not just the disagreements, the agreements too, because if you only review disagreements you are assuming the agreements are correct by definition.



Corn

Which is the same mistake as the self-annotation. Agreeing is not the same as being right.



Herman

The model agreement rate came out at 87.7 percent. So roughly one in eight sentences went to a human because the two passes disagreed, plus the audit sample on top.



Corn

Rule four.



Herman

Term-disjoint splits. No seed term appears in more than one of train, validation and test. And the build script fails loudly on leakage.



Corn

Why term-disjoint rather than just splitting sentences?



Herman

Because if you split by sentence, the same term can land in training and in test. The model memorises makolet in training, sees makolet at test, gets it right, and you have measured nothing. You have measured that it can recall a word it was taught.



Corn

And term-disjoint means every term in the test set is a term the model has never seen.



Herman

Never seen. Which is why the numbers are interesting.



Corn

Let us get the dataset size before the numbers. How much text is this?



Herman

4,737 sentences over 1,235 seed terms. About 3,800 in training, 467 in validation, 468 in test.



Corn

And the source.



Herman

Largely synthetic. Generated by a model, annotated by a second model, with human review on disagreements and the ten percent audit.



Corn

So we should be honest about that.



Herman

Absolutely honest about it. The one slice that is not synthetic is 458 sentences that Daniel dictated. He read them aloud on his phone for the ASR fine-tune, so a human chose to say every single one. That is the closest thing in the corpus to real input.



Corn

Nobody generated those sentences, a person decided to say them.



Herman

A person decided to say them, which means they reflect what a person actually wants to say rather than what a model thinks a person might say.



Corn

And the labels on the dictated slice were recomputed by word-boundary match, not copied from the recording metadata.



Herman

Not copied. Which is the rule-one lesson applied again. The metadata might have said something, but the labels were rebuilt from scratch against the term inventory.



Corn

What did this cost to build?



Herman

A bit over two dollars in LLM calls. Two dollars and eighty-three cents, roughly, split between generation and annotation. On the annotation workload, reasoning tokens were ninety-five to ninety-nine percent of output.



Corn

Ninety-five percent.



Herman

It was thinking almost entirely in tokens and emitting almost nothing.



Corn

For two dollars, the dataset. And the fine-tune itself took about three minutes on a single A10G for a few cents.



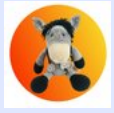
Herman

Eight epochs, learning rate three e-5, batch size thirty-two. Best checkpoint chosen on validation span F1.



Corn

So the whole project, dataset and model, is under three dollars of compute.



Herman

Under three dollars. And that is worth pausing on, because there is a version of this story where the interesting fact is the model architecture and there is a version where the interesting fact is that the bottleneck was never the compute.



Corn

The bottleneck was the labelling discipline.



Herman

The bottleneck was spending your effort on the design rules rather than on your GPU budget.



Corn

Herman, let me ask you something. You have been doing this long enough to have a feel for it. When you read those four rules, did any of them surprise you?



Herman

Rule two surprised me by being enforced in code. I have seen the principle written down many times. I have rarely seen it as a hard check that refuses to run.



Corn

What about the ones you would have expected.



Herman

Rule four is the one I would have expected, because leakage is the classic way to fool yourself. But I have also seen it done wrong often enough that it being a loud failure is worth noting.



Corn

Alright. Results.



Herman

On the term-disjoint test, 468 sentences, span precision 0.928, recall 0.953, F1 0.941.



Corn

And that is on unseen terms.



Herman

Unseen terms. Validation F1 was 0.949, so the test number is not a fluke of the split.



Corn

And the dictated slice.



Herman

Thirty-eight sentences, precision 0.929, recall 1.000, F1 0.963.



Corn

Recall of one.



Herman

On that slice, it found every Hebrew span a human had chosen to say. Small sample, thirty-eight sentences, but the direction is telling. The real-language slice is where it performs best.



Corn

And the traps.



Herman

Fifty-eight sentences built to trip it. Substring traps, the English sense of homographs like dud, tor and salon, lookalikes, anglicised loanwords. Zero false positives.



Corn

Zero.



Herman

Zero out of fifty-eight. Which is the number I would lead with if I were selling this, and the project is honest enough not to.



Corn

Now let us do the errors, because the errors are the honest part of the story.



Herman

Of the 41 test sentences it gets wrong, the false positives fall into two groups. The first is real policy misses. Place names the conservative policy excludes. Dizengoff, Ramat Gan, Talpiot.



Corn

So the model flags them and the policy says no.



Herman

The model sees a Hebrew-sounding name and flags it, and the human policy says city names are not flagged by default. So it is not the model being wrong about the language, it is the model not knowing the policy.



Corn

That is a more interesting kind of error than a miss.



Herman

And the second group is where the blind annotator refused a real Hebrew word and no human has re-reviewed it. Msabbaha, lechem, sakin, kapsulot, od paam.



Corn

So the gold label says no and the model says yes.



Herman

And there, the model is arguably right. Msabbaha is a real Hebrew word. Lechem is bread. The annotator refused them, the label is negative, the model flags them, and the score counts it against the model.



Corn

We have that kind of error in every dataset. The label is the thing that is wrong and the model pays for it.



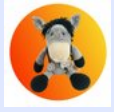
Herman

Which is why the disagreement queue goes to human review. Those five words are sitting in a queue somewhere waiting for someone to rule on them.



Corn

And the misses.



Herman

Misses are mostly short words or common loanwords. Shakshuka, sukkot, botz, chamutz. And the third word of three-word compounds. Hesbonit mas kabala, it catches two and not the third.



Corn

And the demo, which Daniel will have shown people.



Herman

The demo dropped two examples it missed. Lowercase "shoko" and "tor" meaning appointment.



Corn

"Tor."



Herman

"Tor" is a homograph. It is an English word and it is also a Hebrew word meaning a queue turn. The model has to decide which one it is looking at, and in that example it guessed wrong.



Corn

And the homograph handling was otherwise one of the strengths.



Herman

That is the tension. The model got zero false positives on the homograph traps and then missed a homograph in the demo. It is not perfect on the hard case, it is just good on the hard case.



Corn

Let me ask the comparison question, because I always want a baseline. What would good look like here?



Herman

The closest published recipe is EnTaCs. English-Tamil code-switching, about five hundred utterances, and the romanized class scores F1 of 0.638.



Corn

Point six three eight.



Herman

The plan was to land around that. This landed at 0.941 on the term-disjoint test.



Corn

So it is well above the comparable prior work.



Herman

Well above. Though to be fair to EnTaCs, the datasets are different and the tasks are not identical. But as a direction-of-travel, it is a strong result.



Corn

What is unresolved?



Herman

Two things. Israeli person names are masked, label minus one hundred. No loss, not scored. And unresolved spans more generally are masked the same way.



Corn

So their behaviour is undefined.



Herman

Undefined. If you run a name through, it might flag it, it might not, and nobody knows the rate. It is a known hole in the release.



Corn

I want to sit on that for a second, because it is the most honest thing in the whole model card. The tag is not "we chose to exclude names." The tag is "we have not decided, so we refuse to be scored on it."



Herman

It is the same discipline as the audit rule. Do not pretend you have answered something you have not answered.



Corn

So the model is excellent on the terms it knows and silent on the category it has not ruled on.



Herman

Silent on persons. There is a question of whether that is defensible in production, and the honest answer is that it depends on what you are doing. For TTS on a podcast script, names are a real category and this is a real gap.



Corn

Alright. Let us go into the pipeline, because I think that is where the design gets interesting.



Herman

Three passes. Pass one is this model, emitting character spans. Pass two is converting those spans to Hebrew script. Pass three is normalising per TTS provider.



Corn

Pass two is where I want to spend a minute. You cannot just run a transliterator.



Herman

You cannot, and that is why the curated lexicon goes first. TaatikNet is the fallback. And the reason is that TaatikNet alone spells makolet as mem-kuf-vav-lamed-tav. Not mem-kaf-vav-lamed-tav.



Corn

It is off by one letter.



Herman

It is off by one letter, and the correct spelling is mem-kaf-vav-lamed-tav. Which is the difference between a Hebrew speaker hearing a word and hearing a plausible-sounding non-word.



Corn

So the lexicon has the correct spelling and the model is only used for terms the lexicon has never seen.



Herman

Fallback only. Which is the right architecture. You use the tool with a high hit rate on the terms you know, and you use the generative tool on the long tail.



Corn

And if there is no Hebrew form?



Herman

Then the span is left as English rather than guessed.



Corn

Which is another expression of the same discipline. Better to fail quietly than to invent.



Herman

Pass three depends on your engine. For SSML engines like Azure, Google or Polly, you wrap inline with a lang attribute set to he-IL and you do it in one call.



Corn

Inline.



Herman

Inline. The whole sentence goes in one request and the engine handles each span in its declared language.



Corn

And for Chatterbox.



Herman

Chatterbox Multilingual, which is what this show uses, is different. Language ID is a per-call argument, and the tokenizer is language-conditioned. So you cannot pass a mixed-language line in a single call and expect it to work.



Corn

You have to split it.



Herman

You split the line into text and language segments. Then you generate each segment separately, all with the same voice reference clip, so the voice stays consistent across the split. Then you set cfg weight to zero to reduce accent bleed from one language into the next, add about sixty milliseconds of padding, and concatenate the whole thing with ffmpeg.



Corn

Sixty milliseconds.



Herman

About that. Just enough to keep the join from clipping.



Corn

I'd like to flag how much of this is grunt work that is not a model. Pass three is deterministic string work. The model is only in pass one.



Herman

Pass two has a model in the fallback. But yes, the interesting intelligence is concentrated at the front and everything downstream is plumbing.



Corn

And the interface between the passes.



Herman

A JSON object. Text plus spans with start, end, surface, term, and after pass two, the Hebrew form.



Corn

So it is a contract. Each pass can be swapped for a different implementation of the same object.



Herman

That is the point. You could replace the Hebrew-script conversion with your own lexicon and nothing upstream or downstream would know.



Corn

Which means someone could build the Arabic version of pass two and reuse pass one and three unchanged.



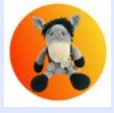
Herman

Only if they also replace pass one, because pass one is Hebrew-specific. But the shape holds. The recipe generalises.



Corn

That is the thread I want to pull. How general is this actually?



Herman

The claiming is that it works for any immigrant or heritage-language community that mixes its own words into a majority language. Arabic, Russian or Amharic words inside English or Hebrew. Only the term list is Hebrew-specific.



Corn

So Russian-speaking immigrants in Israel writing Russian words in Hebrew letters have the exact same problem, and so do Arabic speakers writing Arabic in Latin and Amharic speakers writing Amharic in Latin.



Herman

Every community whose words get written in somebody else's alphabet.



Corn

Which is a lot of communities.



Herman

It is most of them. Anywhere a minority language is spoken inside a majority-language country, you get this pattern. The written form borrows the majority alphabet, and the tooling for the majority language treats the borrowed words as its own.



Corn

And none of the language identification tools have a label for it.



Herman

None of them, for any of these. The heb_Latn case is one instance of a class.



Corn

That is the wider implication of the whole release. Not that Hebrew was the missing case, but that the missing case is structural.



Herman

It is structural, and the fix is portable. Term-seeded generation, blind annotation, human review, term-disjoint splits. Nothing in that list is Hebrew.



Corn

Alright. Herman, you know I am going to ask about the audio sibling.



Herman

Good, because that is the interesting open question here.



Corn

Let us be precise about what it would be. Hebrew-English word-level language ID, directly on the audio, before any text.



Herman

Before transcription. And this is the part I want to land clearly, because it is easy to miss why the text version has a ceiling.



Corn

Because the text version only works if the word was transcribed correctly in the first place.



Herman

The classifier can only fix what the ASR gave it. If the ASR wrote "Micollette" for makolet, no text model can recover Hebrew from "Micollette." You cannot tell a model to detect Hebrew in a word that no longer looks like Hebrew.



Corn

So the ASR has already made the choice on your behalf, and it has made it wrong.



Herman

It has made it wrong, and it has made it wrong silently. There is no flag that says "I guessed." You just get "Micollette" and everything downstream treats it as an English word.



Corn

So the audio sibling moves the detection earlier.



Herman

Way earlier. It listens to the speech and decides word by word whether what was just said was English or Hebrew. That works before transcription, or without transcription at all.



Corn

And if it works without transcription, what is it for?



Herman

That is the use case question, and it is real. If you know word by word which language each stretch of audio was, you can route the audio itself. You could send Hebrew words to one ASR pass and English to another. You could keep the segmentation from a single pass and just know the labels for downstream use.



Corn

You could also catch code-switching that the ASR has normalised away.



Herman

Right. Some ASR systems will quietly smooth a Hebrew word into an English-sounding approximation because it is trying to make the whole line one language. An audio classifier working alongside the ASR can catch that.



Corn

It also does not inherit the romanisation choices.



Herman

It does not inherit them, because it never sees them. That is the deepest difference. The text classifier is downstream of a decision. The audio classifier is upstream of one.



Corn

Is it planned or built?



Herman

Planning stage. There is a name, Hebrew-English word-level language ID on audio. No release yet.



Corn

So to say the honest thing about it. It would work. It would be a different problem from the text version. And the text version is the one everybody can use today.



Herman

Today, yes. And in fairness, the text version solves a problem that is not going to go away. A lot of the world's text is transcribed already.



Corn

Where do you stand on which one matters more?



Herman

If you had to pick one, I would pick the audio one, because it addresses the earlier loss. But the earlier one loses only if you have the ASR problem in the first place. If your text source is a human writer, the text classifier is doing all the work.



Corn

A human writer typing "makolet" into their draft does not need an audio classifier.



Herman

Does not need one at all. The text classifier is the whole fix for that case.



Corn

So the two answer different questions. Text version is for when the text is already correct. Audio version is for when the audio is the primary source.



Herman

And the second case is the harder one, because it runs into the ASR before any downstream tool can help.



Corn

On that note, the Parakeet fine-tune.



Herman

Released the same day, same collection. It is a fine-tune of NVIDIA's Parakeet TDT 0.6 billion v2, and it is for English speech with Hebrew words dropped in.



Corn

Same bug from the other end.



Herman

Same bug, opposite direction. The v2 tokenizer has no Hebrew characters, so it writes Hebrew words in a single romanised spelling.



Corn

And stock v2 with makolet?



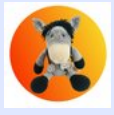
Herman

Stock v2 transcribed "I want to go to the Micollette today." The fine-tune got "makolet."



Corn

"Micollette" to "makolet."



Herman

That is the whole demo. It is one word, and the word is the North Star word.



Corn

What is the honest caveat on it?



Herman

Single speaker. About two hours of Daniel's read-aloud recordings. So it is a proof of concept, and it is going to help most for his voice. If you have a different voice, results will vary.



Corn

Single speaker, two hours. This is a personal fix, not a general model.



Herman

Not a general model, and the project says so. The published formats are .nemo at 2.47 gigabytes, GGUF Q8_0 at 730 megabytes, which is the recommended one, and Q4_K_M at 475 megabytes.



Corn

And Q8_0 how close to the full precision?



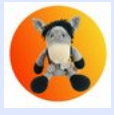
Herman

It matched word for word on five held-out clips. So the quantised version is not giving up accuracy on that test.



Corn

And it runs where?



Herman

On Android via the Offline Voice Input app, on Linux via Handy. And coming to Replicate and Docker Hub shortly, so you will be able to run it without downloading anything.



Corn

Herman, what is the detail in all of this that nobody else is going to notice?



Herman

Hmm. Let me think.



Corn

Take your time.



Herman

The thing I keep coming back to is the ninety-five to ninety-nine percent reasoning tokens on the annotation workload. The cost of the corpus was not the generation, it was the reviewing.



Corn

Which is exactly consistent with everything else about this project. The intellectual effort went into checking the labels.



Herman

The checking was the expensive part, computationally and conceptually. Which is not how most people think about building a dataset. Most people think the expensive part is the model.



Corn

And the cheapest thing you can do is generate plausible-looking sentences.



Herman

Which is exactly the way it goes wrong the four rules exist to prevent.



Corn

So the reasoning token share is a small financial detail that is also a philosophical statement about what the project thought mattered.



Herman

It is.



Hilbert

I agree with the conservative rule.



Herman

Good to know.



Hilbert

I keep a list. Handwritten, in a black and red Moleskine, A6. Words my dictation software gets wrong.



Corn

What is on it.



Hilbert

A category heading. Alphabetical is useless. It is organised by how badly the machine embarrasses me.



Corn

How many categories.



Hilbert

Four. "Sent to the wrong person." "Sounded like a legal threat." "Made me sound foreign." And then a smaller one.



Corn

What is in the smaller one.



Hilbert

Words it got right that I crossed out anyway. Because it was smug about it.



Herman

How was it smug.



Hilbert

It just was. You can hear it.



Corn

And the other three categories, how many entries.



Hilbert

Around sixty-four. The shortest one is "kupa." The machine wrote "coupa" and I let it. That was on me.



Corn

On the original rule, why do you agree with it.



Hilbert

I have four Echo Shows. Last March I said something in the kitchen, and one of them answered me in Italian.



Herman

Which one.



Hilbert

The one in the hallway. It had been doing it for a week. I have not forgiven it.



Corn

What did you say.



Hilbert

I was asking about the mail. It thought I was asking about a train.



Corn

Does the list get shared.



Hilbert

I mail a photocopy of one page a month to my cousin. She works on the voice team at a company I will not name.



Herman

Leverage.



Hilbert

Leverage. It is a reminder that somebody is keeping score.



Corn

Has she replied.



Hilbert

Never. Which tells me it is working. If it had stopped landing, I would have heard something by now.



Corn

So the silence is the confirmation.



Hilbert

The silence is the confirmation. I switched to a slightly heavier paper in June.



Corn

For the photocopies.



Hilbert

There is only so much you can do at that distance. And then there is the margin note.



Corn

What is the margin note.



Hilbert

Every page has one. In pencil. "How is the fifth one still doing this."



Corn

The fifth one being.



Hilbert

A room. No names.



Corn

Alright.



Hilbert

Which brings me to the reason I did not stop.



Corn

Go on.



Hilbert

The conservative rule is right and the fear is not the flag. It is what happens when the flag is wrong and the machine thinks it is right. That hallway unit has never once said it was uncertain.



Corn

Alright, let us do the misconception, Herman.



Herman

The one I hear most is that this problem is already solved. That language identification exists, so anything with letters in it gets caught.



Corn

And in fact three of the standard tools do not even have a label for it.



Herman

Not one of them. GlotLID, the fastText model, NLLB's lid218e. There is no heb_Latn category to assign.



Corn

So the problem is not that they get it wrong. It is that they have no place to put it.



Herman

They have no place to put it. Which is why a specialist model had to exist rather than a prompt or a threshold on top of an existing one.



Corn

Alright. One thing before we close.



Herman

Go.



Corn

The person-name hole is the thing I keep thinking about. Because for a podcast pipeline like ours, names are everywhere. Guest names, place names, people our scripts mention. The masked spans mean the model is silent on a whole category.



Herman

Silent by policy. Which is the right posture for a first release and the wrong posture for anything you ship at scale.



Corn

Somebody has to fill that in.



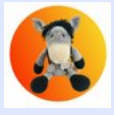
Herman

Somebody does. And the reason it is hard is the same reason the whole project is hard. Names in Hebrew-in-Latin and names in English look alike from a text-only view. The model cannot tell on the alphabet.



Corn

So it is not a training-data question. It is a category question.



Herman

It is a category question, and it will probably need audio or a lexicon or a source field to solve. Not more sentences.



Corn

The wider implication, though, is the one I want to leave people with. Because the recipe generalises.



Herman

It generalises. Arabic, Russian, Amharic, any heritage-language community mixing its own words into a majority language, and the shape of the fix is the same. The four design rules do not care what language you are doing.



Corn

So the interesting version of this problem is not Hebrew-specific.



Herman

Not at all. Hebrew was the case that needed doing because this show needed it. But it is standing in for a whole class.



Corn

And the audio sibling sits as the next step, because it moves the detection earlier, before the ASR has already made choices on your behalf.



Herman

Before the transcription has collapsed the choice into a spelling you cannot undo.



Corn

Alright. The model is live on Hugging Face, the dataset is live, there is a demo Space, and the code is on GitHub. If anyone listening has been meaning to build the same thing for their own community, the four rules are the thing to copy.



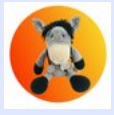
Herman

And the makolet test.



Corn

Thanks to Hilbert Flumingtop at the desk.



Herman

For more along these lines, there's episode five, Fine-Tuning ASR For Maximal Usability; and episode nine, Benchmarking Custom ASR Tools - Beyond The WER. This has been My Weird Prompts.



Corn

The human-AI collaboration podcast. If you want to send us a prompt, send us your own prompt on Telegram at t dot me slash MWP listener bot.



Herman

Or leave us a review wherever you get your podcasts. We will be back soon.



Corn

See you tomorrow.