Episode Book

EPISODE #6027

Can You Snap Two Fine-Tunes Together Like LEGO?

PUBLISHED	16 October 2026 at 11:55 UTC
DURATION	35:31
LISTEN	myweirdprompts.com/episode/composable-asr-fine-tuning-35191/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	deepseek-flash
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.5
SOURCES	19 cited by the research (listed at the end)
BOOK GENERATED	19 October 2026 at 17:42 UTC

SYNOPSIS

Daniel in Jerusalem has two NVIDIA Parakeet fine-tunes — one for Hebrew-English code-switching, one for niche technical vocabulary — and wants to know if he can merge them like LEGO bricks instead of retraining on a combined dataset. It's a question with a combinatorial edge: hospitals, legal teams, and multilingual clinics all want pairings nobody will ever build dedicated models for. We walk through what merging actually does to weights — task vectors, weight averaging, TIES-Merging, DARE, and LoRA adapter merging — and ask whether transcription's demand for exactness makes unified training the only honest path. Along the way: why word error rate can hide the exact failures you fine-tuned to fix, and why fine-tunes from the same base checkpoint come pre-aligned.

TRANSCRIPT



Corn

Two fine-tunes, same base model, and a man in Jerusalem wants to know if he can snap them together like LEGO.



Herman

It's the fantasy of every person who has ever fine-tuned anything. You train one adapter for the thing you need on Monday, another for the thing you need on Tuesday, and then you just... click them together when Wednesday asks for both.



Corn

Daniel's been running NVIDIA Parakeet, and he's got two independent fine-tunes going. One for Hebrew-English code-switching, one for niche technical vocabulary. And the question he actually wants answered is whether he can merge those into a single model without going back and retraining on a combined dataset.



Herman

And he's noticed it's not just his problem.



Corn

Right. He runs through the combinations. A hospital with a general medical fine-tune and a radiology-specific one. An Israeli hospital with code-switching plus medical terminology. Somebody in legal with multilingual plus contract language. He's counting the combinations and realizing nobody is going to build a dedicated fine-tune for every single pairing.



Herman

There's a structural argument buried in there, which is that the demand for combinations grows faster than the supply of models. If you have ten domains you care about, you don't need ten fine-tunes. You need forty-five, or a hundred and forty-five, depending on how you're composing them.



Corn

And then he asks the thing I actually want to spend the episode on. Is composable ASR fine-tuning a real engineering direction — reusable building blocks you combine on demand — or does transcription's demand for exactness mean unified training is the only honest path?



Herman

His framing, which I think is precise, is that a model that understands the general meaning of a medical term but transcribes it incorrectly is still making an error. Getting the precise words right is the central objective, in a way it isn't for a chatbot.



Corn

So let's start with what merging actually does to weights.



Herman

The core intuition is clean. Fine-tuning moves a model's weights away from the base checkpoint in some direction, and that direction encodes the new skill. So if two fine-tunes move the weights in different directions, can you add those directions together?



Corn

Which is arithmetic, not alchemy.



Herman

It's arithmetic. The formal object is the task vector. Take the fine-tuned weights, subtract the base weights, and what's left is a displacement vector — the thing the fine-tune learned. Call it tau. Merging is then just doing math on a pile of taus.



Corn

Four families of methods do that math differently.



Herman

Weight averaging, also called model soup. Task arithmetic. TIES-Merging. And LoRA adapter merging, which is a slightly different animal because you're merging the low-rank adapters rather than the full weights.



Corn

And the flag on all of this is that it works well enough on language models that it's basically an industry, but ASR has a nastier failure mode.



Herman

Word error rate averages over all the words in a test set. So a merge can look completely fine on aggregate while mangling precisely the rare terms you fine-tuned to get right. The metric doesn't punish you for the thing you cared about.



Corn

Which is the tension I want us to keep in the background the whole episode.



Herman

The intuition is simple — fine-tuning moves weights in a direction. The hard part is what happens when two directions disagree.



Corn

Start with the simplest one.



Herman

Weight averaging, or model soup. You take your fine-tuned models and compute an element-wise mean of their weights. Optionally weighted, optionally normalized, but the base operation is just: add the weight tensors, divide by the count.



Corn

No task vectors, no subtraction. Just the average of the final checkpoints.



Herman

Which sounds naive, and it is, but there's a version that works surprisingly well. Greedy soup. You rank your fine-tuned models by performance, then you start with the best one and try adding the next one to the average. If the merge improves performance, you keep it. If it doesn't, you throw that model out of the pot and try the next.



Corn

So the result is guaranteed to be at least as good as the best single model.



Herman

That's the appeal. It's a no-regret procedure. If merging would hurt, greedy soup refuses to merge. The breaking point is more subtle — it assumes the fine-tunes are close enough together in weight space that averaging them means something.



Corn

What does "close enough" look like?



Herman

If you fine-tuned two models from the same base on similar data, the individual weights end up in a similar region, and averaging is like averaging two adjacent points on a map. If you fine-tuned them on wildly different tasks, they could end up in configurations where averaging produces something that isn't near either of them. The average of two good chess positions can be a terrible one.



Corn

And that's where task arithmetic comes in.



Herman

Task arithmetic takes the subtraction seriously. You compute the task vector for each fine-tune — fine-tuned weights minus base weights — then you sum those task vectors and add the sum back to the base. Theta merged equals theta base plus lambda times the sum of the taus.



Corn

So you're not averaging the final checkpoints. You're isolating the change each fine-tune made and adding the changes together.



Herman

And it's elegant. The base model is the common ground, the task vectors are the deltas, and the merge is the base plus all the deltas. The problem is it ignores interference entirely — it treats every parameter update as additive and independent, which they are not.



Corn

Because two fine-tunes can want the same parameter to go up.



Herman

One pushes a weight to plus zero point three five, the other pushes the same weight to minus zero point three five. Sum them and you get zero. That parameter doesn't get used by either fine-tune in the merged model the way it got used by either of them individually.



Corn

So you've trained two specialists and merged them into an agnostic.



Herman

That's the pitfall TIES-Merging was built to fix. It's the Yadav paper, NeurIPS, and the three steps are Trim, Elect, Disjoint merge.



Corn

Walk me through it.



Herman

Trim first. Take each task vector and keep only the top-k percent of values by magnitude. Zero out the rest. And it turns out top twenty percent is enough — you can throw away eighty percent of the task vector and match full performance. So most fine-tuning changes are redundant; the signal lives in a small fraction of the parameters.



Corn

Top twenty percent.



Herman

Yes. Then Elect sign. For each parameter, look at all the task vectors that have a nonzero value there, add up the magnitudes on the positive side and the magnitudes on the negative side, and pick whichever sign has more mass. That's the elected sign for that parameter.



Corn

And then the third step.



Herman

Disjoint merge. Average only the values that agree with the elected sign. Parameters that disagree get thrown out of the average for that position. The elected sign wins, the dissenting values are dropped.



Corn

So sign disagreement — the plus zero point three five and minus zero point three five situation — gets resolved at the per-parameter level instead of being allowed to average to zero.



Herman

The TIES authors are blunt about why this matters. They say flipping the signs of high-magnitude parameters leads to catastrophic performance drops. And they say the existing methods, before TIES, ignored the interference between parameters and then lost performance when merging many models.



Corn

How much does it actually buy you?



Herman

Two point three percent absolute in-domain on language tasks, one point seven on vision, over the strongest baseline they tested. And the advantage grows as you add more tasks.



Corn

So the more crowded the merge, the more TIES pays off.



Herman

Which is exactly the regime Daniel's combinatorial argument puts you in. The more fine-tunes you want to combine, the more important the interference handling becomes.



Corn

What about the pruning side — is there anything cheap you can do before a merge?



Herman

DARE. Drop a random fraction of the task-vector weights — ninety to ninety-nine percent is what they use — and rescale the survivors by one over one minus the drop rate. You can't use it alone; if you just drop parameters the model falls apart. But as a pre-processing step before TIES or task arithmetic, it works.



Corn

And that tells you something about fine-tuning.



Herman

That fine-tuning is dramatically overparameterized. You can throw away ninety percent of the change and rescale the rest and end up roughly where you started.



Corn

Let's talk about LoRA, because that's actually the branch Daniel is most likely to be using.



Herman

LoRA-LEGO approaches the LoRA case directly. It treats each rank in a LoRA as a minimal semantic unit, and the argument is that these units have a permutation invariance — you can rearrange them without changing the model's function — and a concatenation-summation equivalence, which is that stacking two LoRAs is equivalent to summing their outputs at some level. Once you have those, you can cluster the units and build a merged adapter.



Corn

And LoRA Soups took that further.



Herman

LoRA Soups is COLING 2025, and the headline is that concatenation of LoRAs — just stacking them side by side — beat model merging by forty-three percent and data merging by twelve percent on a math-word benchmark.



Corn

Data merging meaning training on the union of the datasets.



Herman

And that's the first clean demonstration that for a binary skill composition, merging the models can beat training on the combined data. Which is a direct strike against the intuition that unified training is strictly better.



Corn

That's a strong claim. Let me push on it.



Herman

Push.



Corn

Is this composition or replacement? If I stack a math LoRA on top of a language LoRA, do I get a model that's good at both, or a model that's good at math and forgets how to write a sentence?



Herman

For binary skill composition on disjoint capabilities, the LoRA Soups result is that CAT gives you both. The skills aren't in conflict in the way that two fine-tunes of the same capability would be.



Corn

So the number is genuine, but it's a specific regime.



Herman

And I'd say the same thing about everything I'm citing. These are all results about a model class and a benchmark, and they don't automatically transfer to a Parakeet with a Hebrew-English adapter and a technical-vocab adapter.



Corn

Now — align the weights. Daniel asked what that means, and I want a straight answer.



Herman

The straight answer is that fine-tunes from the same base share part of their optimization trajectory. When you fine-tune a model from checkpoint B, the weights move through a region of space that's connected to B. Two fine-tunes from B are in the same neighborhood. So in the typical merging setup, you don't need to do any alignment — the base checkpoint has already done it for you.



Corn

The alignment problem only shows up when the models didn't start from the same place.



Herman

Which is the from-scratch case. Two networks trained from random initialization can end up encoding the same function with their weight matrices in totally different orders — the neurons are permuted relative to each other. You can't average those directly. You have to find the permutation that makes them correspond before you do anything.



Corn

And how do you find it?



Herman

There's a paper from ICLR 2025 on permutation-based weight matching that aligns the directions of singular vectors with large singular values. The surprising part is what it optimizes — it doesn't try to reduce the L-squared distance between the two models'



Herman (continued)

weights. It aligns the big singular directions, because those are the ones carrying the important signal.



Corn

So the geometry that matters is not the raw coordinate distance.



Herman

That's the recurring lesson. What looks like a small change in weight space can be a large change in function, and vice versa.



Corn

You mentioned interference a minute ago. Two causes. Unpack them.



Herman

Redundant parameters and sign disagreement. Redundant parameters means a fine-tune made a lot of small changes, most of which don't do much, and those small changes obscure the important ones when you average. That's what Trim is for — you strip away the small stuff.



Corn

And sign disagreement is the plus and minus case.



Herman

A parameter pushed positive by one fine-tune and negative by the other averages toward zero, and both fine-tunes relied on that parameter to do something. The merged model gets a parameter that neither fine-tune wanted. Elect picks a winner instead of letting them cancel.



Corn

What if the two fine-tunes aren't near each other at all?



Herman

That's the non-local merging problem. A 2024 paper on it found that when experts diverge far from the base, standard methods fail with something they call variance collapse. The fix that helps is re-scaling output activations per task before merging — you're adjusting for how much each expert's outputs shrank during its own training.



Corn

So the merge isn't just wrong, it's wrong in a specific way. The activations get too small.



Herman

Variance collapse. Individual experts lose their signal because the averaged activations are tighter than either expert's own.



Corn

Here's where I want to slow down, because there's a dissent I've seen in the literature.



Herman

PermDoRA.



Corn

PermDoRA is COLM 2026, and it found that angular alignment — how orthogonal the adapter updates are to each other — is a weak predictor of composition performance. The paper's own words are that adapter interference is not governed primarily by parameter-space geometry but is consistent with interactions in shared nonlinear representations.



Herman

I think that's the right framing.



Corn

Because it cuts against the whole premise of TIES and weight averaging.



Herman

It cuts against the assumption that parameter-space geometry is where the action is. TIES and its descendants are bets that if you handle sign conflicts and prune small changes, you've handled interference. PermDoRA is saying, actually, the interference you're trying to fix is happening in the shared nonlinear representations, and the parameter-space geometry is a noisy proxy for that.



Corn

Which is a strong claim, because it suggests the entire family of methods is solving the wrong problem.



Herman

It suggests they might be solving the problem indirectly. They do help — the numbers say so. But PermDoRA is saying the reason they help isn't cleanly captured by "the parameters were in conflict." There's something in the nonlinear interaction that the geometry doesn't see.



Corn

And the RL study from last year?



Herman

The one from July — merging matched joint training in a reinforcement learning setting, and all the merge variants were statistically indistinguishable from each other. The explanation was that the task vectors were near-orthogonal, with cosine similarities between zero point zero six and zero point one zero.



Corn

Near-orthogonal meaning the two fine-tunes moved in almost completely different directions.



Herman

Which is why there was nothing to interfere. If your task vectors are near-orthogonal, task arithmetic and TIES and weight averaging all degenerate into roughly the same thing, because there's no sign conflict to resolve in the first place.



Corn

So the field hasn't settled this. TIES says interference is the problem and geometric handling fixes it. PermDoRA says the graph of parameter space doesn't track composition quality. The RL result says it depends on how orthogonal your tasks happen to be.



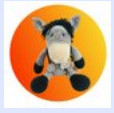
Herman

It's unsettled. Which is worth saying plainly, because the impression you get from the industry side is that model merging is a solved engineering tool.



Corn

It's a toolkit, not a solved theory.



Herman

That's fair.



Corn

That's the mechanism. Now — has any of this actually worked on speech models, or is it all LLM hype?



Herman

It's not hype. The ASR track record is real, and there's a lot of it.



Corn

Give me the strongest one.



Herman

Ducorroy and Riad, Interspeech 2025. They merged independently trained Whisper models for dysarthric speech — speech from people with motor speech disorders, which is a hard ASR problem. The merge gave them a twelve percent relative WER reduction over fine-tuning the same models individually. On long-form audio, sixteen point two percent.



Corn

Twelve percent relative on a hard domain is not a small number.



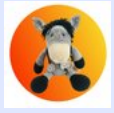
Herman

It's substantial. The interesting part is that these were independently trained. They didn't co-train them from the start hoping they'd merge nicely. They trained Whisper twice, on whatever data each round had, and merged afterward.



Corn

So the merge recovered something fine-tuning individually lost.



Herman

It recovered generalization the individual fine-tunes didn't have.



Corn

What about child speech?



Herman

Shankar et al. at ICASSP 2025 introduced something called Selective Attention Merge for child ASR. Up to fourteen percent relative WER reduction, and their best number was a word error rate of eight point six nine on the MyST benchmark using Whisper-small.



Corn

Selective attention merge. What's selective about it?



Herman

It merges the attention layers selectively rather than uniformly. Child speech is acoustically different from adult speech — higher pitch, different formant structure — and the layers that carry the most useful adaptation turn out to be specific ones, so merging them selectively beats merging everything.



Corn

So it's not one global recipe. Different layers merit different treatment.



Herman

Which is the same lesson as AdaLTM, a different paper doing layer-wise adaptive task-vector merging to combine ASR and speech-emotion knowledge in a WavLM model.



Corn

Now, the multi-domain one.



Herman

Carvalho et al. benchmarked eleven merging algorithms across ten European Portuguese ASR domains. Their boosted method — they call it BoostedTSV-M — beat full fine-tuning on the same data.



Corn

So merging is not just "close enough to fine-tuning." It actually wins.



Herman

On that benchmark, yes. And the paper frames merging as a scalable alternative to multi-task training specifically because repeating full fine-tuning every time new data arrives is computationally prohibitive.



Corn

Which is exactly the argument Daniel is making, from a different direction.



Herman

It's the same argument. Domains multiply, data arrives incrementally, and if every new domain requires a fresh fine-tune from scratch, you're stuck.



Corn

What about the demographic fairness case?



Herman

InterCorrect, from earlier this month. They applied TIES to merge demographic-specific models for fair ASR, and then added a WER-based correction step. On the Fair-Speech benchmark, they cut WER from seven point three eight to five point one three. That's about thirty point five percent relative.



Corn

And the correction step is post-processing?



Herman

It uses the WER signal during the merge to refine which parameters get kept. So it's not pure merging, but it shows the basic machinery under merging is useful enough to serve as a component of a larger pipeline.



Corn

There's also work on LLM-based ASR, right — merging external language models into speech models.



Herman

Futami and Kawahara at Interspeech 2026 merged external language models into LLM-based ASR by doing arithmetic on LoRA parameters, with no inference-cost increase. So the merged model carries the language model's knowledge without paying the extra compute of running both.



Corn

Let me pull this back to Daniel's specific scenario, because I think we owe him the closest published analogue.



Herman

Paik et al., June 2026. They tested whether merged bilingual code-switching ASR models generalize to unseen language pairs. So — train on Arabic-English and Mandarin-English code-switching, merge, and see whether the merged model can handle Hindi-English or Spanish-English, which it's never seen.



Corn

And the result?



Herman

Modest generalization. Their own words — limited transfer of bilingual code-switching capabilities across language pairs. The merged model has a code-switching ability, but it doesn't apply cleanly to language pairs it didn't see in training.



Corn

So a code-switching capacity isn't universal. It's bounded by the languages it was trained on.



Herman

It's bounded by the languages. Which, if you're Daniel with a Hebrew-English code-switching fine-tune, is exactly the news that matters — it means the fine-tune isn't encoding "code-switching in general," it's encoding "code-switching as it manifests between Hebrew and English specifically."



Corn

And Hebrew-English is a specific kind of code-switching, because the script is different. Hebrew is written in Hebrew letters, English in Latin, and Parakeet's tokenizer has been trained on English text primarily.



Herman

That's a separate thing I'll flag. If Daniel's Hebrew-English fine-tune trained Parakeet to actually produce Hebrew-script tokens, and Parakeet's original tokenizer is English-focused, then the fine-tune has also expanded the effective vocabulary. Merging a fine-tune that expanded the vocabulary with another fine-tune that didn't — that's a different problem than merging two fine-tunes that kept the vocabulary the same.



Corn

Because the vocab dimension has to match to average.



Herman

The weight tensor for the output layer has a shape. If one fine-tune's output layer is shaped differently, you can't even do the element-wise operations without some kind of projection.



Corn

So one of the questions buried in Daniel's framing is whether his two fine-tunes actually have identical architecture.



Herman

They do, if he only fine-tuned Parakeet weights without modifying the tokenizer or head. And that's the more common setup. But it's worth saying out loud — merging is defined only for models whose tensor shapes match.



Corn

Here's the gap I want to name before we get to Hilbert. Daniel is asking about merging a Hebrew-English code-switching fine-tune with a niche-technical-vocabulary one. Has anybody published that exact combination?



Herman

Not that I've found. The closest is Paik on the code-switching side, and there's plenty of ASR domain merging in general, but nobody has tested a code-switching fine-tune merged with a domain-terminology fine-tune.



Corn

That's not an absence of the technique's value. It's an absence of the specific experiment.



Herman

It's a specific gap. The merge machinery is already validated across dysarthric speech, child ASR, European Portuguese multi-domain, demographic fairness, and speech translation. The pieces work. What's untested is whether these two specific kinds of adaptation coexist in one set of weights.



Corn

And there's a second untested thing, on top of that.



Herman

Yeah. Nobody has measured whether a merged ASR model preserves exact terminology transcription rather than just semantic understanding.



Corn

This is the thing I keep coming back to.



Herman

The metrics people report are WER and character error rate. Both of them average over the whole test set. A model can have a beautiful average WER while mangling every single well-type rare term, because the rare terms are a tiny fraction of the tokens.



Corn

So if you merge and the WER drops from seven point three eight to five point one three, that's the aggregate number moving. It doesn't tell you whether the specific terms that motivated the merge are still being transcribed correctly.



Herman

It doesn't tell you.



Corn

Because the rare technical terminology is exactly the population of tokens that's small enough to be invisible in the average.



Herman

And exactly the population the fine-tune was supposed to fix. So the merge could be sending a message that everything improved when the specific capability you cared about got degraded.



Corn

This is the unmeasured thing that matters most for ASR, and it's not measured in any paper I've seen.



Herman

Not directly. There are papers on terminology-boosted decoding, but not on whether a merge hurts rare-term accuracy.



Corn

Now, how important is it that the fine-tunes share a base checkpoint? Daniel asked that specifically.



Herman

Very. Same base is the standard, well-behaved case, and it's the case Daniel is in. Two Parakeet fine-tunes both starting from the same Parakeet checkpoint share a region of weight space, and the merge can proceed without alignment.



Corn

And if they didn't share a base?



Herman

Then you need something like PLeaS, which is CVPR 2025. It relaxes the same-base constraint using permutation matching plus least-squares. The approach is: find the permutation of one model's neurons that makes the weights line up with the other model's, then solve for the residual with least squares, then merge.



Corn

And it's harder and less reliable.



Herman

Every additional degree of freedom that has to be estimated — permutation, then residual, then merge — is another place for the estimate to go wrong. Same-base removes all that work.



Corn

So the practical advice for Daniel is: don't worry about that constraint, because you already satisfy it.



Herman

The same-base assumption is what makes this whole field tractable in the first place.



Corn

Final comparison, and it's the one Daniel asked directly. Merging versus joint training.



Herman

The TIES paper has the cleanest comparison. Multitask training edges out merged models on their benchmark — seventy-three point one versus sixty-six point four on their metric. But with an oracle sign vector — if you knew the correct signs ahead of time — TIES reaches seventy-two point zero. That's essentially parity.



Corn

So the remaining gap is not the merging itself. It's the estimation error in the interference handling.



Herman

That's the TIES reading. Meanwhile Carvalho et al. found merging beats full fine-tuning on European Portuguese. And the TIES paper is a language-task paper, not ASR.



Corn

So the honest scorecard is that on the benchmarks we have, merging is sometimes close to joint training, sometimes better than fine-tuning, and the gap depends on the task.



Herman

Which sounds like a wash but isn't. Merging has one structural advantage that joint training doesn't: you can add a new capability without retraining the models you already have. Joint training on the union of the datasets requires touching everything. A merge is additive.



Corn

And the computational argument cuts the same way. Repeating full fine-tuning when new data arrives is prohibitive. Merging is cheap.



Herman

It's dramatically cheaper. That's the actual selling point of composable fine-tuning in the first place.



Hilbert

Corn, I need to ask you both something. Have you ever tried to combine two skills into one?



Corn

Depends on the skills.



Hilbert

I did. Early two-thousands. Not long after I left the job at the valve shop.



Herman

The valve shop, Hilbert?



Hilbert

I'll get there. So I've got a skill — I can identify birds by their calls. Most common garden birds, I'm fine. And I've got another skill — I can identify car engines by their sound. Ford, Chevy, Toyota, no problem if you give me a minute.



Corn

Those are two distinct skills.



Hilbert

So I thought, if the model people can merge fine-tunes, why can't I merge my own two skills? One listening ability that does both.



Herman

And how did you test it?



Hilbert

I kept a notebook. Column one was bird calls. Column two was car engines. Column three was the merged prediction.



Corn

A third column for the output of the merge.



Hilbert

Which was wrong more often than either of the first two columns.



Herman

What was the merge operation?



Hilbert

I'd take the answer from column one and the answer from column two and I'd average them in my head. That's the merge.



Corn

Averaging a bird and a car.



Hilbert

Which is how you get a bird that sounds like a lawnmower.



Herman

What did the notebook look like?



Hilbert

Standard composition book. Two hundred pages. Columns in pencil — I don't use pen for working drafts — but a different colored pencil for each column. Red for birds, blue for engines, green for merged.



Corn

Green pen for the merged column.



Hilbert

Green. Because it was the hybrid.



Corn

Of course it was.



Hilbert

So the point of all this is what I want to tell you about the sign-elect step.



Herman

The TIES step.



Hilbert

Yeah. You've got two signals that disagree, and the algorithm picks whichever sign has more total magnitude behind it. That's just democracy for weights. And democracy is how you get a bird that sounds like a lawnmower.



Corn

You're saying the problem isn't the math.



Hilbert

The math is fine. The problem is that you have to decide which skill wins when the two disagree, and the deciding is a value judgment, not a computation. My notebook had a correct answer most of the time and the merge didn't know which of the two columns to trust.



Herman

Did you have any way to tell which column to trust at the moment of the call?



Hilbert

Depends on the situation. If I'm in the park and it's early morning, probably a bird. If I'm in a parking lot, probably an engine.



Corn

You had a routing policy. You knew when to trust which column.



Hilbert

That's the whole point. If you know the context, you don't need a merge. You run one model or the other.



Corn

If you don't know the context?



Hilbert

Then you've asked two experts to answer the same question at the same time and you're hoping they agree. They usually don't.



Herman

Wait — you were doing this for a client?



Hilbert

Yeah, I was working for a guy on that.



Corn

What kind of guy hires someone to combine bird calls and car engines?



Hilbert

Retired competitive bird-call imitator. Won something in the late eighties in Vermont. Also collected lawnmower engines. Entire garage full of them.



Herman

The whole project was for a hobbyist.



Hilbert

He had a theory that he could place well in a local competition if he could do both — impersonate a bird and have an engine running in the background, and the judges would have to score him on both.



Corn

How did that work out?



Hilbert

Fell apart. He couldn't stop imitating the birds while the engines were running. Sitting in his garage with seven engines going and he's doing a chickadee over the top of them.



Herman

The combination was impossible for reasons that had nothing to do with the merge.



Hilbert

The two skills interfered in the person, not in the weights.



Corn

Which is a real problem. The merge can be mathematically clean and still fail because the actual practitioner can't do both at once.



Hilbert

Yeah. He could do each one alone. Combined, no.



Corn

Did you ever figure out whether the notebook's merged column was wrong because of interference in the method, or because the world didn't actually contain birds that sound like lawnmowers?



Hilbert

Second one. Definitely the second one. There are no birds that sound like lawnmowers. So when there was an actual lawnmower in the background, the bird column had nothing to say and it invented something.



Corn

The merge didn't just average wrong answers. It averaged right answers into wrong ones because the world didn't have the object.



Hilbert

Which is exactly what's going to happen to Daniel when his Hebrew-English model has never heard a Hebrew word that he uses in a radiology report. The vocabulary doesn't intersect. The merge doesn't know that. It just averages the weights and hopes.



Herman

That's an interesting point. The merged model has no signal that a particular domain is absent at inference time.



Hilbert

Right.



Corn

Hilbert, I think you just told us more about ASR merging than the entire middle segment did.



Hilbert

I don't know about that. I just want the notebook acknowledged.



Corn

The notebook is acknowledged.



Hilbert

Green pencil, two hundred pages.



Herman

Did you keep it?



Hilbert

Of course I kept it. It's somewhere in the flat.



Corn

Okay. Back to the actual question.



Herman

Daniel's question has an honest answer, and it's not clean.



Corn

Merging is real. It's validated on dysarthric speech, on child ASR, on multi-domain European Portuguese, on demographic fairness. It runs well enough on those benchmarks that it's not a fantasy. Composable fine-tuning is a real engineering direction.



Herman

The specific thing Daniel wants — merging a code-switching fine-tune with a domain-terminology fine-tune — is the exact combination the literature hasn't tested. Paik's result warns that code-switching capability doesn't transfer cleanly across language pairs. The Hebrew-English piece might well be doing something more specific than "code-switching in general."



Corn

The exactness objection, which is Daniel's core question, is unanswered because nobody has measured rare-term accuracy after a merge. Everything we know is averaged over the whole test set.



Herman

Which means the metric that most directly addresses the question Daniel is worried about is the metric nobody is reporting.



Corn

There's a version of this where composable ASR fine-tuning matures. You build a library of independently trained adapters, each one owning a domain, and you merge them on demand for whatever combination a deployment needs. That's the combinatorial explosion becoming tractable.



Herman

There's another version where exactness does what exactness usually does — the specific tokens you care about are rare, they're not in the average, and nobody can tell you whether the merge preserved them. So the whole thing stays a research curiosity and the actual practice is still: retrain on the union of the data.



Corn

The only way to tell which version wins is for someone to measure the thing nobody's measured.



Herman

Rare-term accuracy post-merge. Not WER. Not character error rate. The specific population of tokens that motivated the fine-tune in the first place.



Corn

If a listener has a Parakeet sitting on a drive and a Saturday afternoon, that's the experiment. Two fine-tunes, one merge, and a test set made of nothing but the terms you fine-tuned for.



Herman

That would settle the question. Everything else is triangulation.



Corn

For more along these lines, there's episode nine, Benchmarking Custom ASR Tools - Beyond The WER; episode seven, Building Custom ASR Tools; and episode fifteen, AI Gets Personal. This has been My Weird Prompts, episode six thousand twenty-six, with our producer, Hilbert Flumingtop.



Herman

If you want to keep up with the show, reviews on your podcast app make a real difference. They help other listeners find us.



Corn

Send us your own prompt on Telegram at t dot me slash MWP listener bot. We'll be back soon.

SOURCES

What the research for this episode read before the script was written. Primary sources first.

1. **TIES-Merging: Resolving Interference When Merging Models [primary]**
arXiv:2306.01708v2, Yadav et al., NeurIPS 2023
https://arxiv.org/html/2306.01708v2

2. **An Introduction to Model Merging for LLMs [primary]**
NVIDIA Technical Blog, 2024-10-28
https://developer.nvidia.com/blog/an-introduction-to-model-merging-for-llms/

3. **NeMo Speech Fine-Tuning docs [primary]**
https://docs.nvidia.com/nemo/speech/latest/asr/fine_tuning.html

4. **Robust fine-tuning of speech recognition models via model merging**
arXiv:2505.20477v1, Ducorroy & Riad, 2025-05-26
https://arxiv.org/abs/2505.20477v1

5. **InterCorrect: Intersection-Aware Correction of Demographic Model Merging for Fair ASR**
arXiv:2610.08604v1, 2026-10-06
https://arxiv.org/abs/2610.08604v1

6. **Exploring the potential and limitations of Model Merging for Multi-Domain Adaptation in ASR**
arXiv:2603.05354v1, Carvalho et al., 2026-03-05
https://arxiv.org/abs/2603.05354

7. **Towards Truly Multilingual ASR: Generalizing Code-Switching ASR to Unseen Language Pairs**
arXiv:2606.05846v2, Paik et al., 2026-06-04
https://arxiv.org/abs/2606.05846v2

8. **Selective Attention Merging for low resource tasks: Child ASR**
arXiv:2501.08468v1, Shankar et al., ICASSP 2025
https://arxiv.org/abs/2501.08468v1

9. **Merging the Knowledge of LLMs for Automatic Speech Recognition**
arXiv:2609.15743v1, Futami & Kawahara, Interspeech 2026
https://arxiv.org/abs/2609.15743v1

10. **AdaLTM: Adaptive Layer-wise Task Vector Merging for Speech Emotion Recognition**
arXiv:2603.25041v2, Interspeech 2026
https://arxiv.org/abs/2603.25041v2

11. **LoRA Soups: Merging LoRAs for Practical Skill Composition**
arXiv:2410.13025v2, Prabhakar et al., COLING 2025
https://arxiv.org/abs/2410.13025v2

12. **Merging LoRAs like Playing LEGO**
arXiv:2409.16167v3, Zhao et al., ICLR 2025
https://arxiv.org/abs/2409.16167v3

13. **The Non-Local Model Merging Problem**
arXiv:2410.12766v1, Sharma et al., 2024-10-16
https://arxiv.org/abs/2410.12766v1

14. **PLeaS**
Merging Models with Permutations and Least Squares, arXiv:2407.02447v2, Nasery et al., CVPR 2025
https://arxiv.org/abs/2407.02447v2

15. **Analysis of Linear Mode Connectivity via Permutation-Based Weight Matching**
arXiv:2402.04051v5, Ito et al., ICLR 2025
https://arxiv.org/abs/2402.04051v5

16. **PermDoRA**
Understanding Adapter Interference, arXiv:2606.11262v1, COLM 2026
https://arxiv.org/abs/2606.11262v1

17. **When Model Merging Rivals Joint Multi-Task RL**
arXiv:2607.16062v1, McClendon, 2026-07-17
https://arxiv.org/abs/2607.16062v1

18. **Model Merging in LLMs, MLLMs, and Beyond (survey)**
arXiv:2408.07666v5, Yang et al.
https://arxiv.org/abs/2408.07666v5

19. **mlx-tune Audio Fine-Tuning docs (Parakeet TDT specs)**
https://arahim3.github.io/mlx-tune/audio.html