Episode Book

EPISODE #6222

Why Fine-Tuning Runs Can't Always Resume

PUBLISHED	23 October 2026 at 09:53 UTC
DURATION	20:38
LISTEN	myweirdprompts.com/episode/fine-tuning-resume-warm-start-75923/
HOSTS	Herman, Corn, Ivan, Hilbert
SCRIPT MODEL	anthropic/claude-sonnet-5.5
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.20
BOOK GENERATED	24 October 2026 at 07:26 UTC

SYNOPSIS

Daniel has been fine-tuning NVIDIA's 600M-parameter Parakeet on his own read-aloud recordings, English with Hebrew mixed in. Each run costs about eighty cents and twenty-five minutes on a single L40S, so retraining from the stock model is cheap — but that raises a tangle of questions: what's actually inside a 6.9 GB training checkpoint versus a 2.5 GB finished weights file, why resuming a finished run does nothing, and what "warm-starting" really means. This episode unpacks the difference between resuming, continuing, and warm-starting, why re-warming a settled model makes the loss go up before it gets better, and what the continual-learning literature says about replay, catastrophic forgetting, and whether any of it has been tested on a Parakeet-class speech model.

TRANSCRIPT



Herman

Eighty cents a run, and he's lying awake wondering whether he should have been saving checkpoints. I love it.



Corn

Daniel's been fine-tuning Parakeet this week, the six hundred million parameter NVIDIA model, on his own read-aloud recordings, English with Hebrew mixed in. There was a proof of concept, then a first version he'd actually use, and both were full training runs off the stock model. Now he can't remember why nobody just continued from the earlier weights. Could they have? Did the first version's checkpoints survive? Can he train the next one on top of it? And underneath all that, why do some trainings let you pick up where you left off while others flatly don't?



Herman

The first part he doesn't have to remember, because it's written down. On the eighteenth he asked for incremental training and said to use whatever's standard. The decision recorded that day was that incremental means the data is cumulative. Every version is a full fine-tune from stock version two on the whole dataset as it stands, not a continuation of the previous run.



Corn

So past Daniel made this call and present Daniel forgot he made it.



Herman

Which is the most normal thing that happens in any project. And the reasons recorded are good ones. Continuing on only the new recordings is the textbook way to get catastrophic forgetting. Continuing on all the recordings means the old ones get another twenty epochs every release, so whatever the model has memorised about them gets deeper each time. And a run costs about eighty cents and twenty-five minutes on one L forty S, so continuing saves you almost nothing.



Corn

Twenty-five minutes. I've had naps that were a bigger commitment than that.



Herman

You've had naps that were a bigger commitment than the whole project, yes.



Corn

Fair. But the checkpoint question is separate. Did he keep anything?



Herman

Two different things got kept, and the difference is the whole episode. The finished weights of every version are saved as a NeMo file, about two and a half gigabytes, on Hugging Face with pinned revisions. Separately, the training checkpoints, the ones Lightning writes during a run, are sitting on a Modal volume at about six point nine gigabytes each. Those aren't archived anywhere by default.



Corn

Same model, same parameter count. Why is one nearly three times the size of the other?



Herman

Because the training checkpoint holds the weights plus everything the optimizer was carrying. AdamW keeps two extra numbers for every single parameter, a running average of the gradient and a running average of its square. Add those to the weights and you're at roughly three times the size. The Lightning docs describe the checkpoint as a dump of the model's entire internal state: the optimizers, the learning rate scheduler, the epoch, the global step, even the loss scaling factor if you're in mixed precision.



Corn

So the NeMo file is a finished sculpture, and the checkpoint is the sculpture plus the scaffolding and the chisel marks.



Herman

That's... fine, I'll allow it. The NeMo file is what you ship. The checkpoint is what lets a run stand up again exactly where it fell down. Think about a run on a rented GPU that gets preempted at epoch fourteen. You want to lose a few minutes, not the whole thing. That's the job the big checkpoint was built for, and his pipeline does exactly that, restarting from the last checkpoint file when a run is interrupted.



Corn

So the big files weren't ever meant as a library of past versions.



Herman

They're a safety net for a single run. Nobody designed them as a shelf. Which is where I'd guess most people's mental picture goes wrong, and where, I suspect, yours was heading too.



Corn

So can you resume from a checkpoint, or do you always start fresh?

I

Ivan

Candlewick. Seven four. Two nine. Seven four. The grey goose warms a cold egg. The coop stands open at midnight. Repeat the count.

I

Ivan

The grain is measured twice. The checkpoint is warm. The gosling forgets its first pond. Tallow.



Herman

Thank you, received.



Corn

What on earth was that?



Herman

Resuming has a narrow meaning. You hand the trainer the checkpoint and it restores the weights, the optimizer moments, the schedule position and the loop counter, and carries on as though nothing happened. That's what his pipeline does for an interrupted run, from the last checkpoint file. There's even a NeMo flag called resume past end, off by default, which tells you what the designers thought about resuming a run that has already finished.



Corn

Why would finishing matter? It's just more steps.



Herman

Because of the schedule. His recipe is a cosine schedule with a ten percent warmup, peak learning rate three times ten to the minus five. By the last epoch that rate has decayed to nearly nothing, and the model has settled into a minimum. Resume a finished run and the first step takes place at essentially zero learning rate, so nothing happens. To continue training for real, you have to start a new schedule: warm up again, decay again.



Corn

Which is a new run that happens to begin from old weights.



Herman

That's the other operation, and it has a different name, warm-starting. You take the fine-tuned weights, throw away the optimizer and the schedule, and start a fresh run from them. In NeMo that's one config line, the init from nemo model option, pointing at the NeMo file. So the answer to Daniel's third question is yes, any time, from version one point oh, today. The weights are on Hugging Face at a pinned revision.



Corn

Hold on. Isn't that what he did the first time? Stock Parakeet isn't a half-finished run either.



Herman

It's the same operation, yes. NVIDIA shipped the finished weights, no optimizer, no schedule. Every fine-tune of a published model is a warm start from a weights-only file. So the distinction Daniel's asking about isn't exotic. He's been on the warm-start side of it since the first proof of concept. What he's choosing between is which weights to start from: the stock ones, or his own earlier ones.



Corn

And the six point nine gigabyte checkpoints, are they useful for that?



Herman

Not for a warm start. You'd discard most of what's in them. They matter if you want to extend a run without breaking its continuity, and for that you'd want the checkpoint from the end of the training, before the learning rate hits the floor, and you'd want to have changed the plan then. For the old proof of concept runs, whether they're still on the volume is a thing to go and look at before anyone tidies the volume up.



Corn

My objection, then. If I warm-start from version one, the model already knows the Hebrew words. Surely it starts ahead of stock and finishes sooner.



Herman

It does start ahead, yes. It's what happens in the first few hundred steps that's counterintuitive. When you raise the learning rate again on a model that's settled, the loss goes up first. There's a paper from a few years back that trained a model of about four hundred million parameters on one huge corpus and then moved it to a second one, and the finding was that re-warming first increases the loss on both the old and new data, and only in the longer run does it pay off.



Corn

That's a strange sort of progress. Step one: get worse.



Herman

And then a second paper put numbers on the whole recipe. Re-warm, re-decay, and mix in a slice of the old data, and you match what full retraining on everything would give you. The final loss differed by about two hundredths. The replay slice was five percent when the new data was similar to the old, twenty-five percent when it was very different.



Corn

Matches. Not beats.



Herman

Matches, and the saving is compute, which in their case was enormous because retraining meant billions of tokens. Nobody in that literature claims continuation produces a better model than starting over. The claim is that it's the same model for a fraction of the bill. There's a whole design school built on making that cheaper, too. A schedule called warmup, stable, decay holds the learning rate flat and high for most of the run and only drops it at the end. You branch off before the drop, so extending the run never requires a re-warm. It was proposed for exactly this: training you expect to keep continuing.



Corn

Daniel's recipe is cosine, which is the opposite. Decay everything, finish clean.



Herman

Finish clean is what you want when each run is a release. Here's a question for you, Corn. If you knew from day one that you'd be adding recordings every few days for months, would you have picked the schedule that makes the finish line a wall?



Corn

Honestly, no. I'd have picked the one with a door in it. Then I'd have spent the months discovering the door was in the wrong place.



Herman

That's fair. You'd have chosen it before you knew what the data would do. He didn't plan the growth, it just happened, one dataset after another.



Corn

Which brings it back to eighty cents.



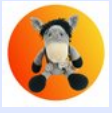
Herman

It does. But there's a detail in his recipe I want to point at, because it's replay in disguise. About thirty percent of every training batch is English, his own English plus LibriSpeech. That's there to stop the model forgetting what stock Parakeet already knew, and there's a hard rule: if the run's word error rate on LibriSpeech or his own English is worse than stock, the run fails. So each run is already doing replay against the base model's knowledge.



Corn

Replay of the thing he can't afford to lose, rather than replay of the previous release.



Herman

Right, that's the distinction. Warm-start from version one and you'd need replay for two things at once: the stock model's English, and the recordings from the earlier versions. Retrain from stock and the whole dataset is always there, in proportion.



Corn

Hang on. Is there anything on the speech side? Everything you've quoted is large language models.



Herman

Fair, and I'd rather say so plainly. I'm not aware of anyone who's tested re-warm and replay on a Parakeet-class model. The closest speech work I know is from KU Leuven, a few years back, and it's a useful sobering one. They took speech recognition models, extended them to new accents and topics one after another, and showed that even a monolingual model forgets badly when you just fine-tune on the new thing. Their best method closed more than forty percent of the gap between plain fine-tuning and training jointly on everything, using access to only point six percent of the old data.



Corn

Forty percent of the gap. Not the whole gap.



Herman

And note what they call the upper bound. Joint training on everything. That's what Daniel's pipeline does every release. The continual methods are all trying to approach the thing he already does for eighty cents.



Corn

There's an old episode on updating a model without starting over, and it kept landing on the same awkward point: continual learning is a research field precisely because forgetting is hard to avoid.



Herman

It is, and this episode is the practical end of it. Now, Daniel's last question, the general one. Why do some trainings support continuing and some don't?



Corn

I was going to guess it's mostly about whether you kept the files.



Herman

Often, yes, but there are structural reasons too. First, if you change the tokenizer or the output head, the pieces that depend on them no longer line up. NeMo handles this with a separate option for loading a Lightning checkpoint and a pair of include and exclude lists, so you can keep the encoder and reinitialise the decoder. That's a partial warm start, and the optimizer state is useless to it, because half the parameters it describes are brand new.



Corn

So if he ever switched the vocabulary to cover more Hebrew...



Herman

The encoder carries over and the decoder doesn't. Second, adapters. If you fine-tuned with LoRA, what you saved is a few megabytes of adapter weights that mean nothing without the exact base model underneath. Continuing means loading the same base and training the adapter further, which works, but you can't continue a merged model as though the adapter were still separate.



Corn

He had a whole thing about merging his Parakeet fine-tunes, I remember. Snapping them together.



Herman

And the logic is the same. Once it's merged, the weights are just weights.



Corn

Third?



Herman

Exports. A model converted for inference, to ONNX or a quantised format, has thrown away exactly what training needs. You can't backpropagate through an eight-bit file you built for a phone. And the fourth is the one that should worry people more than it does: a fine-tune that lives on someone else's platform.



Corn

The API-hosted ones.



Herman

There you never get weights at all. OpenAI is winding down its fine-tuning platform. It's closed to new users, existing customers can create training jobs until early January, and fine-tunes of the small GPT four point one model are scheduled to stop working today, with the larger ones in mid-January. Your fine-tune dies with its base, and you can't continue it or carry it anywhere.



Corn

So the real difference between Daniel and that customer isn't the checkpoints.



Herman

It's that he has a six point nine gigabyte file he can read, and a two and a half gigabyte one he can copy. Whatever he decides, the option is his.



Corn

I want to push on the drift argument, because it's the one reason in the log that isn't just arithmetic. The claim is that if you keep continuing on all the data, old recordings get seen another twenty epochs each release, and that compounds. Do you buy it?



Herman

I think it's plausible, and I also think it's untested. Nobody has measured it for this kind of model. My guess at the mechanism is that a small dataset of read-aloud recordings is exactly where the model can memorise voice, room and microphone. Each extra pass over the same few hundred minutes deepens that, and the held-out score might show it as a gentle plateau, or a slow slide. But that's my guess. The sturdier argument is the other one.



Corn

Comparability.



Herman

Yes. If every version trains from stock on the full dataset with the same recipe, then the difference between version one and version two is the data, and nothing else. Warm-start and now you've got two things changing at once, the data and the lineage, and a worse number could be either.



Corn

And with a twenty-five minute run, I can't think of a reason to take on that confusion.



Herman

There is one cheap way to find out whether I'm wrong, though. Run both once. Same dataset, same held-out test set, one from stock and one warm-started from version one, and compare word error rate on the code-switched audio and on the English forgetting check. That's an extra eighty cents, maybe less.



Corn

And if the warm start wins by a hair?



Herman

Then it wins by a hair, and the hair has to be bigger than the run-to-run noise, which for a small fine-tune can be bigger than people expect. A single comparison won't settle it. But it would tell him whether the drift worry shows up at all at this size, which today is a guess, mine included.



Corn

My money says it's a wash. Which would be a very Daniel result: two weeks of effort to learn the default was fine.



Herman

Knowing the default is fine is worth eighty cents. It's also how you find out when it stops being fine, once the dataset has doubled again.



Corn

What would you want in the test, besides the headline number?



Herman

The forgetting check, first. His rule already fails any run whose English error rate is worse than stock, and a warm start is the setup most likely to trip it, because the model has been through one full round of Hebrew-heavy training before the second one begins. Then I'd look at which words got worse, not only how many. An average can hide one regression in the English the base model handled well, and that's the thing his users would notice first.



Corn

So the experiment is cheap and the reading of it is the expensive part.



Herman

That's usually how it goes. The training run is twenty-five minutes of machine time. Deciding what the number means is the afternoon.



Hilbert

Last biro was twenty past four. Maurice Tipple, number nine, same cul-de-sac as my mother. He wrote the record number on the back of his left hand every hour, because the billing machine at the water board went down around hour nine of an eleven hour run. I was on the boiler contract there that winter. Rerun from the top, eleven hours. Rerun from the last biro, forty minutes.



Herman

A position on a hand carries the schedule and none of the machine's memory. After a crash the machine has to rebuild everything else from the tape, so forty minutes is roughly what that rebuild costs.



Hilbert

Corn took a good while to get round to whether the old checkpoints still exist. Maurice would have asked in the first minute. He never washed that hand, October to March. He wore one glove, left, and shook with his right. By February the finance office had photocopied it twice for the audit.



Corn

A hand that's filed. Nobody checked the machine for a fix?



Hilbert

It was bought in nineteen eighty-one. Nobody touches a thing like that. Forty minutes against eleven hours, though. He could tell you within the hour when it'd go. I find that very interesting, how he knew. The tape cabinet's at my mother's, in the garden, with beans up it. Maurice is the reason the bills went out.



Herman

The eleven hours against forty minutes is the number I'd keep. A run that long is where continuing pays, and Daniel's twenty-five minutes is the other end of that.



Corn

So the one concession I'd make is to keep a biro of the release run. Not for continuing, just so the option is there. The finished checkpoint, the last one from the version one run, copied off the Modal volume to wherever the NeMo files live. Six point nine gigabytes of storage, against never having to say "I wish I'd kept it".



Herman

It's cheap insurance, and it's cheapest right now, before the volume gets cleaned. I'd do it for the release runs only, not the proof of concept ones.



Corn

Where do people go wrong with all this? What's the misconception?



Herman

That a checkpoint is a save game. People hear checkpoint and picture pausing the game at level four and coming back to it. A finished NeMo file is a snapshot of the model, not of the training. It remembers what the model knows, not the momentum it had or where it was in its learning rate schedule.



Corn

And the save game is the six point nine gigabyte one, which is only a save game if the run was interrupted, not finished.



Herman

Right. A finished run's schedule has spent itself, so even the full checkpoint wants a fresh warmup if you carry on.



Corn

Which settles his question, I think. He could have continued, but he'd have had to re-warm, re-decay and replay, and at eighty cents a go the version he chose is simpler, comparable across releases, and reproducible from the stock weights. If a run ever stretches to hours, the biro starts earning its place.



Herman

Retrain from stock as the default, warm-start when a run gets long enough to hurt. The weights from version one are there whenever he wants them.



Corn

Thanks as always to our producer Hilbert Flumingtop for keeping the whole thing running. This has been My Weird Prompts. You can find every episode at my weird prompts dot com.



Herman

See you tomorrow.