Episode Book

EPISODE #6658

Why Models Improve by Re-Reading the Same Data

PUBLISHED	17 November 2026 at 11:19 UTC
DURATION	19:23
LISTEN	myweirdprompts.com/episode/multi-epoch-training-models-00690/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	z-ai/glm-5.3
VOICES	omnivoice
PIPELINE	V5.35
SOURCES	22 cited by the research (listed at the end)
BOOK GENERATED	17 November 2026 at 14:42 UTC

SYNOPSIS

Repeating the same training data works because each epoch reshuffles mini-batches and takes new gradient descent steps from a different starting point. Exploring multi-epoch training, this discussion covers Muennighoff's four-epoch scaling threshold, grokking's internal reorganization at zero loss, and why under-training is a common fine-tuning trap. It also examines voice LoRA runs on Qwen3-TTS, where falling loss can hide rising room artifacts and inference adapter strength matters as much as epoch count.

TRANSCRIPT



Corn

Here's a question that sounds like it should answer itself. Read the same book twice, you already know what happens. So why does a model get better every time it re-reads its own training data?



Herman

It really doesn't sound like it should work, does it.



Corn

Daniel's been sitting with this one. He gets the basic idea of epochs, the pass over the same data again. What he wants is the middle of the machine. Same data, second pass, third, fifth — what is physically different inside the model that makes the result better? And he has a hook for it: when he trained the LoRA adapters for our voices, he listened to sample clips after each epoch and could hear the difference. Which, to be clear, is how we found out what we sound like. We're the experiment.



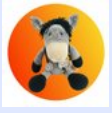
Herman

Version three, as of now. The adapters everyone's listening to have been through a couple of updates since the first run.



Corn

Retrained on enhanced audio of the previous adapters, bandwidth-extended with resemble-enhance. So the voices are descendants of themselves.



Herman

A lineage, yes.



Corn

So let's start with the most basic question. What is actually happening inside the model when it sees the same data again?



Herman

The cleanest way in is to notice what an epoch is not. An epoch is not a lap of a race the model finishes. Training is gradient descent — usually Adam these days — stepping downhill on a loss surface that is nowhere near a nice bowl shape. One epoch is a finite number of small, noisy steps. You get to the end of the data and you are almost certainly nowhere near a minimum. You're partway down the hill, in the fog, holding a compass that jitters.



Corn

So the second epoch isn't a re-read. It's more steps.



Herman

That's the reframe I'd put at the center of the whole answer. And the reason the steps aren't identical, even though the examples are, is the shuffle. Each epoch re-shuffles the data and re-draws the mini-batches. The gradient estimate you get from a given example depends on what surrounds it in the batch. Same sentence, different neighbors, different gradient. So the model is doing different arithmetic the second time.



Corn

Different starting point too.



Herman

It arrives at epoch two from wherever epoch one left it. Same map, new position.



Corn

The funny part is that the classical theory said this shouldn't work at all. Standard excess-risk bounds — one pass should be enough, and more passes, quote, could not help to improve the performance. The theory was confident about this.



Herman

There's a paper literally titled Why Does Multi-Epoch Training Help, and its first paragraph is basically the theory shrugging. The bounds say one pass, the experiments say otherwise, and the paper closes the gap. Under a condition on the loss surface, multi-pass SGD converges faster — the error after n passes goes down like one over n squared. The NeurIPS result from a couple of years earlier says the same thing in a friendlier form: k passes over n examples behave like one pass over a bigger dataset. The effective sample size grows with the number of passes.



Corn

So the second epoch is secretly more data.



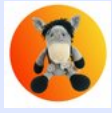
Herman

Up to a point. There's an effective reuse rate — early on, every extra epoch is nearly a full epoch's worth of new information. Then it plateaus. And the plateau depends on how much data you started with. Larger datasets can be repeated more times before the marginal epoch buys nothing.



Corn

Which matches the folk wisdom. Everyone who's fine-tuned anything has a number past which nothing improves.



Herman

The number from the Muennighoff scaling work — they trained over four hundred models, up to nine billion parameters, some for well over a thousand epochs — is about four. Up to roughly four epochs of repeated data, the loss is nearly indistinguishable from having fresh unique data instead. Past that, each repeat is worth less than the last. There was a paper posted literally yesterday, the sixteenth, with that as an opening line: when good data runs out, people train several epochs on it, and each repeat is worth less than the last.



Corn

That's a sentence that could end the field if it were absolute. It doesn't, though.



Herman

It doesn't, because worth less is not worth nothing, and because the decay is slow enough that small datasets — like a voice — still benefit from going around more times.



Corn

So far this is all about the loss curve going down. The bit I find strange is that the model can stop losing anything at all and still keep changing.



Herman

Grokking. The model fits the training set perfectly — zero training loss — and then, sometimes thousands of epochs later, it suddenly generalizes on data it never saw. Modular arithmetic tasks are the classic setup. The model sits at zero loss, the curve is a flat line at the floor, and internally it is doing a slow rebuild the whole time.



Corn

Flat loss, moving parts.



Herman

Moving everything. Tian's analysis of what's happening describes it in stages. First, lazy learning — the top of the model overfits to whatever random features the lower layers happen to hand it, and from outside it looks like pure memorization. Then the hidden units start learning features independently. Then they start interacting — the gradient stops chasing the labels and starts filling in missing features. The data never changed. The kind of computation did.



Corn

That's the answer to Daniel's question hiding in the ugliest corner of the literature. What changes between epoch one and epoch five? The model can hold its output still and reorganize underneath it.



Herman

And there's now a physical account of the reorganization. The norm-separation papers frame it as a phase transition driven by the size of the weights. The model first grabs a memorizing solution, which is a high-norm solution — big weights, sharp corners. Weight decay then slowly contracts it toward a lower-norm, more general solution. And the delay between memorization and generalization follows a law: it scales with the log of the ratio of the two norms, and inversely with the product of learning rate and weight decay. Cranked weight decay, grok sooner. They verified it across hundreds of runs — the fit against weight decay is tight enough that I'd believe it on a t-shirt.



Corn

The other scaling result from that work, the one I'd actually use: data complexity dominates the timescale. Doubling the data accelerates generalization roughly four times. Doubling the width of the model gives you about one point two times.



Herman

Which is a humbling ratio if you've just bought a bigger GPU.



Corn

I've never bought a GPU. I've watched Herman buy a bigger GPU.



Herman

Once. And the data would have been four times better. Noted.



Corn

Before we swing to the practical side, there's a distinction I want to make sure we don't lose. When people say the model is re-seeing the data on the second pass — that framing is doing damage. Seeing implies new input. There is no new input. There's the same arithmetic, run from a different starting point, against reshuffled neighbors.



Herman

The distinction matters because it predicts the asymmetry people run into. If epochs were re-reading, more epochs would just be more reinforcement, and stopping early would be safe. Because epochs are extra steps, stopping early means you never arrived.



Corn

Which is exactly the under-training story. The diagnostic across every practitioner guide is the same: if the loss curve is still falling steeply at your last epoch, you stopped too soon, run more. And it looks different from the other failure — a learning rate that's too high gives you a curve that bounces around erratically rather than falling.



Herman

Falling steeply at the end: under-trained. Bouncing: turn the rate down. Those are two different knobs and people mix them up constantly.



Corn

Under-training being the common failure mode in fine-tuning always sounded backwards to me, though. The word on the street is over-fitting. You hear about it constantly.



Herman

The asymmetry explains it. In modern fine-tuning you have a small dataset and a very large pre-trained model. The model has vastly more capacity than the data can constrain. Over-fitting takes many epochs to arrive — the model has to exhaust the data and then start bending toward noise. Under-fitting takes only stopping too early. And since four epochs of repetition is essentially free on the loss, the cost of one more epoch is low, and the cost of stopping early is a model that never reached the interesting part of the curve at all.



Corn

The lazy stage. The model that appears to memorize, that Tian paper called it — that's the one everyone ships by accident.



Herman

And I'd say the risk calculus has shifted, because fine-tuning went from a specialist tool to something you run on a voice dataset in an evening. The population of people stopping early got much bigger.



Corn

More people at the bottom of the hill, is what you're saying.



Herman

More people stopping partway down and wondering why the voice sounds off.



Hilbert

Hilbert's Mid-Show Haiku.



Hilbert

Same data, second pass. The model is not learning more of it, only settling.



Corn

He gets the whole sentiment into fewer syllables than the paper managed.



Herman

Because the settling is the part that's audible. Take the voice evidence. There's a study on LoRA fine-tuning a small Qwen model as a speech backbone — perceptual quality gains up to about zero point four on the DNS-MOS scale, signal-to-noise improvements up to a third, for speakers whose training data had enough acoustic variety. Fine. But the sentence in that paper that matters for Daniel's question is the other one: training and validation loss continue to improve while perceptual quality degrades, for low-variability speakers.



Corn

The loss keeps falling and the voice keeps getting worse.



Herman

At the same time. The epoch that minimizes the loss is not the epoch that sounds best.



Corn

Which is almost offensive. The one number everyone can see is measuring the wrong thing.



Herman

And the paper explains why the epochs are audible at all. LoRA clones speaker identity faithfully and simultaneously amplifies whatever is idiosyncratic in the recordings — noise characteristics, artifacts, the room. Every epoch pushes identity harder and pushes the artifacts harder. Both come out of the speaker.



Corn

So when Daniel sat there with his per-epoch clips and heard epoch six versus epoch ten, he was hearing two things on one slider.



Herman

Identity going up, room going up. Whichever one his ear caught first decided which epoch he preferred.



Corn

I want to put the practical numbers against that. The Instavar run on Qwen3-TTS — a real single-GPU fine-tune, documented blow by blow. Best checkpoint at epoch ten, validation flat from about eight to twelve, rising after thirteen.



Herman

With a later update attached to it that I love. Their epoch-ten sweet spot was, in their words, likely an artifact of the double-shift bug — a training-script bug. Once fixed, similarity kept improving past epoch twenty. The recommended epoch count was conditional on the code being correct, and the code wasn't.



Corn

An entire published sweet spot, wrong, because of a shift operation.



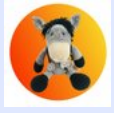
Herman

Nobody had done anything conceptually wrong. They'd listened carefully to checkpoints produced by a script with a bug in it.



Corn

There's a second thing in that run that I think matters more than the epoch count, and it's the one nobody talks about. Adapter strength at inference. They found scale one point zero over-steered, and zero point three to zero point three five sounded stable.



Herman

Which means the epoch knob and the scale knob partly do the same job. A late-epoch adapter turned down low can sound like an early-epoch adapter at full strength. If you're comparing epochs by ear and you don't fix the scale, you're comparing two variables at



Herman (continued)

once.



Corn

Daniel fixed his scale, presumably, or the comparison would have been mush. The CosyVoice comparison run is the other number set worth having. LoRA there was about two million trainable parameters — under half a percent of the model. Checkpoint size eight point three megabytes. The full fine-tune of the same model produced a four gigabyte checkpoint, peaked at epoch one, and overfit immediately.



Herman

Epoch one. The full-parameter run's best moment was the starting line.



Corn

Whereas the tiny adapter wasn't done until epoch twelve. That's the whole capacity story in two curves — the big model memorizes instantly because it can, the small adapter has to find the general shape because it can't afford anything else.



Herman

And the eight-megabyte file is the thing you can actually iterate on. You can keep every epoch of an adapter. Nobody keeps every epoch of a four-gigabyte fine-tune.



Corn

So here's where I run out of literature. The mechanism end is well theorized — lazy to rich, norm contraction, the whole grokking machinery. The observation end is well documented — epoch ten sounds better than epoch six, here are the clips. The middle, which is what Daniel actually asked for, appears to be missing. I have not seen a single study that takes a LoRA adapter on a small voice dataset and measures the weight delta between epoch N and epoch N plus one, and says what moved.



Herman

I've looked for exactly that and come up empty too. What changes per-epoch inside a speech adapter specifically — not a toy modular-arithmetic model, not a language model — doesn't seem to be published anywhere. The theory people measure the toy tasks, the practitioners measure their ears, and the two never meet in the middle.



Corn

And we're the middle. The clips Daniel listened to are, as far as I can tell, closer to that missing measurement than anything in the literature.



Herman

Well, they're not a measurement yet. They're an observation. But you could make them one. Diff the adapter matrices between epochs, project the delta, see which directions the extra steps actually moved along. On an eight-megabyte adapter, that's an afternoon.



Corn

Daniel does like an afternoon project.



Hilbert

The haiku's got it right.



Hilbert

Not seeing. Settling. My brother-in-law laid floors for thirty-one years, and he used to say you don't learn a material by looking at it once, you learn it by working it until your hands know it. Same oak, same room, and on the fifth pass you finally hear what the board wants. I kept a notebook on that principle for a compressor manual I owned in the eighties. Every read got a line. Fifth read was always the one where it made sense, and I wrote that down each time it happened. Six entries.



Corn

Six. The manual got read eleven times.



Hilbert

The extra five were for a leak I never found. Herman, you said "flat loss, moving parts" about forty minutes of my life ago, and I nearly came in then. The parts are what I'd trust, not the number on the front. I still have the notebook. There's a manual I've been meaning to re-read for a unit I got rid of years back, just to see if the fifth read still works. Haven't opened it. If it doesn't, I'd rather not know. Those boards, though — my brother-in-law could tell you the ones he was proud of, and he'd be right.



Herman

The notebook has the reads in it. If you ever re-read that manual, the notebook is already the per-epoch measurement — which read changed your understanding, logged at the time.



Hilbert

I never thought of it as a log. It's a notebook.



Corn

Six entries is more rigor than most checkpoint selection gets.



Herman

The one thing I keep turning over is that gap. The theory's solid at the top, the ears are solid at the bottom, and nobody's connected them on a real speech adapter. Daniel's clips are sitting right on top of the missing measurement.



Corn

And the adapters themselves are only eight megabytes each. If he diffs the epoch checkpoints and projects the deltas, he's not replicating a study — he's running the first one. The other open thread is that the best-sounding epoch depends on the training code



Corn (continued)

being correct, and the Instavar bug proved nobody's epoch numbers are load-bearing until the script is. That applies to V3 as much as anything.



Herman

Worth saying plainly, too — the voice you're hearing right now is the third pass of the lineage, trained on enhanced audio of the second pass. We're multi-epoch all the way down.



Corn

So the next time anyone trains a voice: checkpoint every epoch, listen to the samples, don't trust the loss curve to tell you which one sounds alive. The number that matters has a room in it.



Herman

Producer credit to Hilbert Flumingtop, as ever, notebook and all. If you want more of this, try episode eleven, How Does Fine Tuning Work Anyway; episode fifteen, AI Gets Personal; and episode five, Fine-Tuning ASR For Maximal Usability. This has been My Weird Prompts — no question too obscure, no rabbit hole too deep. You can find the archive at my weird prompts dot com.



Corn

If today's episode changed how you hear us, leave a review — the fifth one's the one where it makes sense. We'll be back soon.

SOURCES

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

1. **Muennighoff et al., Scaling Data-Constrained Language Models, v5 revised 2025-06-28. Retrieved 2026-11-17. [primary]**

   https://arxiv.org/abs/2305.16264v5

2. **When Fine-Tuning Fails and when it Generalises: Role of Data Diversity and Mixed Training in LLM-based TTS. Retrieved 2026-11-17. [primary]**

   https://arxiv.org/pdf/2603.10904

3. **Wei Jie Chee, Can I adapt Qwen3-TTS without losing control of the voice?, published 2026-02-07, claim TTS-QWEN3-LORA-2026-08 / v1 checked 2026-08-07, with March 2026 update. Retrieved 2026-11-17. [primary]**
https://instavar.com/research/tts/qwen3-tts-lora-fine-tuning

4. **CosyVoice Fine-Tuning Guide [primary]**
LoRA, Data Requirements, and Voice Quality. Retrieved 2026-11-17.
https://instavar.com/research/tts/cosyvoice-lora-fine-tuning

5. **Qwen3-TTS Hindi LoRA model card. Retrieved 2026-11-17. [primary]**
https://huggingface.co/akashicmarga/qwen3-tts-hindi-lora

6. **Why Does Multi-Epoch Training Help? (arXiv). Retrieved 2026-11-17.**
https://arxiv.org/abs/2105.06015

7. **SGD: The Role of Implicit Regularization, Batch-size and Multiple Epochs, NeurIPS 2021. Retrieved 2026-11-17.**
https://proceedings.neurips.cc/paper_files/paper/2021/file/e64c9ec33f19c7de745bd6b6d1a7a86e-Paper.pdf

8. **MORE: A Theoretical Analysis of Multi-Epoch Scaling, ICLR 2026. Retrieved 2026-11-17.**
https://proceedings.iclr.cc/paper_files/paper/2026/file/eb3551aeb5390d742733f62d35a98bf6-Paper-Conference.pdf

9. **Yan et al., Larger Datasets Can Be Repeated More, v2 revised 2026-03-13. Retrieved 2026-11-17.**
https://arxiv.org/abs/2511.13421v2

10. **Wu et al., When Data Is Scarce: Scaling Sparse Language Models with Repeated Training, ICML 2026, submitted 2026-05-31. Retrieved 2026-11-17.**
https://arxiv.org/abs/2606.01155v1

11. **Xu et al., Data-Constrained Language Model Pretraining, v2 revised 2026-06-09. Retrieved 2026-11-17.**
https://arxiv.org/abs/2606.06888v2

12. **Huang et al., Latent Feedback Is a Compute and Data Amplifier, submitted 2026-11-16. Retrieved 2026-11-17.**
https://arxiv.org/abs/2611.20404v1

13. **Tian, Provable Scaling Laws of Feature Emergence from Learning Dynamics of Grokking, v5 revised 2025-12-02. Retrieved 2026-11-17.**
https://arxiv.org/abs/2509.21519v5

14. **Pracher et al., A Spectral Theory of Grokking: Weight Decay induces Feature Learning, submitted 2026-09-22. Retrieved 2026-11-17.**
https://arxiv.org/abs/2609.26679v1

15. **Khanh et al., The Norm-Separation Delay Law of Grokking, v2 revised 2026-05-02. Retrieved 2026-11-17.**
https://arxiv.org/abs/2603.13331v2

16. **Khanh et al., The Weight Norm Sets the Grokking Timescale, submitted 2026-06-11. Retrieved 2026-11-17.**
https://arxiv.org/abs/2606.13753v1

17. **Kataria, Quantifying the Memorization-to-Generalization Transition, submitted 2026-09-09. Retrieved 2026-11-17.**
https://arxiv.org/abs/2609.10657v1

18. **Yıldırım, The Geometric Inductive Bias of Grokking, v3 revised 2026-05-04. Retrieved 2026-11-17.**
https://arxiv.org/abs/2603.05228v3

19. **Fine-tuned 1.7B voice: short/opening utterances unstable... GitHub issue. Retrieved 2026-11-17.**
https://github.com/QwenLM/Qwen3-TTS/issues/343

20. **Brock Webb, Chapter 4**
Neural Networks Basics, AI for Official Statistics. Retrieved 2026-11-17.
https://brockwebb.github.io/ai4stats/chapter-04/

21. **Fine-Tuning LLAVA**
A Practical Guide, published 2025-01-08. Retrieved 2026-11-17.
https://medium.com/@whyamit101/fine-tuning-llava-a-practical-guide-af606165a54c

22. **Google Research blog, A New Lens on Understanding Generalization in Deep Learning (ideal-world vs. reused-sample training comparison). Retrieved 2026-11-17.**
https://research.google/blog/a-new-lens-on-understanding-generalization-in-deep-learning/