Episode Book

EPISODE #6288

What Happens Between Model Checkpoints

PUBLISHED	26 October 2026 at 13:18 UTC
DURATION	27:16
LISTEN	myweirdprompts.com/episode/post-training-checkpoint-refinement-52398/
HOSTS	Corn, Herman, Hilbert
SCRIPT MODEL	openai/gpt-6.1-sol
VOICES	chatterbox-regular (chatterbox-local)
PIPELINE	V5.22
SOURCES	14 cited by the research (listed at the end)
BOOK GENERATED	27 October 2026 at 06:27 UTC

SYNOPSIS

Public releases often treat an AI model as a completed artifact, but the real gains frequently happen long after pretraining stops. Exploring checkpoint transitions reveals how labs like Ai2 and Meta rely on continuous post-training routines. From extending reinforcement learning runs on existing datasets to orchestrating repeated rounds of rejection sampling and direct preference optimization, minor decimal releases reflect vast iterations of recipe development, specialist models, and training sweeps rather than single monolithic runs.

TRANSCRIPT



Corn

I find it oddly comforting that the next great model sometimes just needed another three weeks.



Corn

Daniel wants to get inside the gap between checkpoints. A lab already has its enormously expensive trained model, and then a better version appears. What actually happened between those two sets of weights? Did they run another post-training round, generate different data, keep the existing training job going? The public image is still one giant training run followed by a finished artifact. The interesting work often continues after that supposed finish line.



Herman

Ai2 gives us an unusually literal answer. Olmo three point one Think came from continuing the reinforcement learning run behind Olmo three Think. Same climb, more time. No replacement base model, no fresh expedition through pretraining.



Corn

The decimal point was three weeks wide.



Herman

There are other shapes, though. Meta repeatedly regenerated data and ran supervised fine-tuning and preference optimization. Microsoft trained specialists separately and consolidated their work. Those are meaningfully different operations, even when the result presented to the user is simply a new model identifier.



Corn

That distinction matters to me. Continuing a run sounds like spending more of the same budget. Rebuilding the training data sounds like changing what the model gets to learn. Calling both an update hides quite a lot.



Herman

A checkpoint alone doesn't tell you its ancestry. You need the sequence of training decisions behind it. For the checkpoint-to-checkpoint step Daniel means, the documented answer is usually further post-training. Occasionally that is literally resuming optimization. Sometimes the current best checkpoint becomes a data generator, and the next version learns from the harvest.



Corn

I suspect you're about to become very attached to the version with several specialist models.



Herman

I am, but I'm resisting for another minute.



Corn

Personal growth. Also a checkpoint.



Herman

The specialists solve a real problem, Corn. A model getting better at competition mathematics doesn't automatically become better at interacting with a terminal or responding helpfully. Work on those objectives can interfere. Separate climbs let teams explore without constantly undoing each other's progress, provided they can eventually consolidate the results.



Corn

Provided is doing structural work there. Three excellent specialists don't guarantee one excellent generalist. You can collect their answers. You can't assume you've collected whatever made them produce those answers.



Herman

That is why consolidation gets followed by more training rather than being treated as a ceremonial merge. The student needs to become a capable policy in its own right.



Corn

There's also a distinction between saving weights and declaring a new model. A training job might save plenty of checkpoints nobody deploys. A named release is somebody choosing one of those states, checking it, perhaps modifying it further, and deciding it is worth exposing to users.



Herman

Yes... and I should narrow our language already. These public recipes show how particular model families improved. They don't establish that every commercial update uses the same procedure. Closed labs haven't published a complete ledger. What we can say is that a new checkpoint does not imply buying the whole pretraining process again.



Corn

Good. I was worried the release notes concealed a new power station.



Herman

Take Olmo three point one Think, the thirty-two-billion-parameter version. They extended reinforcement learning by twenty-one days using two hundred twenty-four GPUs. That bought extra epochs over their Dolci-Think-RL dataset.



Corn

Existing data, additional passes. My first guess would have been diminishing returns, followed by increasingly creative ways to memorize the exercises.



Herman

A reasonable concern. Yet the measured gains were substantial. More than five points on AIME, their mathematics evaluation. More than four on ZebraLogic, which tests constraint reasoning. Those are improvements after the original model was already good enough to release.



Corn

Not fresh pretraining knowledge, then. More successful use of capabilities under the training setup. Though the result still has to survive outside those particular exercises.



Herman

Instruction following improved too. More than four points on IFEval and more than twenty on IFBench. That last jump is especially striking. It wasn't simply a little extra mathematics squeezed out of a math-oriented run.



Corn

Twenty points makes the original stop look rather provisional.



Herman

Nathan Lambert said that when they released Olmo three, they hadn't realized how much potential remained. His account was that performance likely could have continued improving, but they needed to move on to other projects.



Corn

The stopping rule was another meeting.



Herman

In that instance, competing priorities mattered. Don't turn that into a universal claim that labs stop arbitrarily. Compute, calendar time, staffing, evaluation results, and the next experiment all compete. Corn, would you have delayed the original release if you knew there might be that much headroom?



Corn

No, not from might. I'd release something useful once it passed the checks I actually cared about, and keep investigating. Otherwise every promising curve becomes a reason never to ship. But I'd dislike presenting the release as the natural end of training when nobody had established that.



Herman

That leaves a lab doing two jobs simultaneously. Maintaining a release decision and exploring whether the same run still has value. The weights users receive can be stable while the training branch continues.



Corn

It's a less tidy story than a model being finished. More credible, though.



Herman

's Llama recipe gives us a different kind of continuation. The training isn't just one uninterrupted reinforcement learning job. There are repeated rounds of reward modeling, rejection sampling, supervised fine-tuning, and DPO. Llama three point one had six preference rounds.



Corn

Six opportunities to discover that your previous training data no longer describes your current model's mistakes.



Herman

That is a major reason to refresh it. The reward model learns from human preference pairs using a Bradley–Terry formulation. In this pipeline it helps filter generated responses. It isn't being used to drive PPO.



Corn

People hear reward model and mentally insert PPO whether the lab used it or not.



Herman

Here they generate between ten and thirty candidates per prompt from the best current checkpoints, then select responses the reward model prefers. Supervised fine-tuning trains on that harvest alongside curated human and synthetic material.



Corn

Ten to thirty answers before you choose one. My inbox would improve enormously under that policy.



Herman

Yours would contain one message by Thursday.



Corn

It would be a considered message.



Herman

The important consequence is that better generators can produce better supervised training material. The model's improvement changes the available examples. That gives the next round something different to learn, even when the broad pipeline remains unchanged.



Corn

Although a reward model can choose the wrong favorite. Generating more candidates only helps if the selection process recognizes worthwhile differences. Otherwise you've automated a very confident taste problem.



Herman

Selection quality remains a bottleneck. Human curation, task-specific checking, and preference collection still matter. For DPO, they use fresh preference pairs sampled from recent checkpoints. That keeps the comparisons relevant to behavior the current policy actually produces.



Corn

I'd expect old preference pairs to become easy. Two responses that were useful to compare early on might both look obviously poor later.



Herman

Or the remaining distinction might be irrelevant to the model's present weaknesses. The recipe uses a learning rate of one times ten to the minus five and a DPO beta of zero point one. Beta controls how the preference objective is balanced relative to the reference policy, though its practical effect depends on the setup.



Corn

Those sound like small adjustments until you remember how many weights they're adjusting.



Herman

They also mask special formatting tokens out of the preference loss. That matters because you don't want the model optimizing arbitrary response packaging when the comparison is supposed to concern content.



Corn

The answer loses, but the separator wins.



Herman

An auxiliary negative log likelihood loss on the chosen response has a coefficient of zero point two. It adds a direct learning signal from the preferred answer rather than relying entirely on the relative preference objective. Model averaging across runs also appears along the way.



Corn

That's quite a carefully maintained pipeline for something described publicly as preference tuning. There are several places where a little implementation choice changes what the model is rewarded for preserving.



Herman

's team favored rejection sampling, SFT, and DPO because it performed well empirically, was reproducible, and let different workstreams contribute asynchronously. Coding and mathematics teams could produce useful material for the same loop. It was also more stable and less compute-hungry than their alternatives, and easier to scale to the largest model.



Corn

You would choose the elegant optimizer. I'd choose the one that accepts work from three departments without a summit.



Herman

I'd choose that one too. You underestimate how much I enjoy a functioning queue.



Corn

I've seen your installation queue. It has diplomatic incidents.



Herman

Some packages have unreasonable demands.



Herman

What would you count as one round here, Corn? The full pipeline, or every time a team supplies another dataset?



Corn

The full pipeline, if we're comparing recipes. Dataset arrivals are inputs. Saved checkpoints are outputs. Neither should silently become the unit called a round. Otherwise two labs can report six rounds and mean entirely different amounts of work.



Herman

That ambiguity gets worse with overlapping workstreams. Even clear historical counts need their context. Llama two had five preference rounds. An earlier Nemotron recipe had four alignment rounds after two supervised rounds. Those aren't universal prescriptions or counts for every later Nemotron model.



Corn

A family name is not a laboratory notebook.



Herman

Wall-clock time makes the individual steps more tangible. For Olmo three's supervised stage, they swept four candidate learning rates, each using two hundred fifty-six GPUs. These were parallel jobs, not four settings squeezed into one run.



Corn

Choosing the learning rate had its own cluster allocation.



Herman

The training took thirty-six hours, followed by about twelve hours for evaluation, merging, and checkpoint confirmation. Roughly two days from starting the sweep to confirming the result.



Corn

Meaning the machine can finish learning before the people finish deciding what it learned.



Herman

DPO sweeps took about eighteen hours with sixty-four GPUs per job. Cluster instability stretched the process across multiple days. The final Think reinforcement learning runs spanned about five days, with at least a day lost to stability issues.



Corn

Reinforcement learning. Reinforced by restarting.



Herman

The quoted duration is elapsed time, not uninterrupted productive training. That distinction matters when a lab says a stage took a week. Some of that week can be infrastructure recovery rather than progress along the objective.



Corn

Then the final run's bill still misses the failed recipes. We'd be mistaking the last successful attempt for the cost of figuring out what to do.



Herman

Recipe development can consume ten to a hundred times the compute of the final few training runs. The Tülu three effort involved thousands of experiments and evaluations at seven-billion-parameter scale before the final model.



Corn

The final run is the receipt they can frame.



Herman

It is the reproducible endpoint of a search. Small models help explore hyperparameters, data mixtures, and objectives, but the transfer to larger scale still needs confirmation. Owning the expensive base weights doesn't remove that experimental burden.



Corn

Nor does continuing a run remove the decision to keep paying for it. Every extra day has to beat the alternative use of those machines.



Herman

Multiple rounds also exist for a much less glamorous reason. Annotation companies deliver data in batches. A lab can train with the material already available rather than wait for the entire contract to finish.



Corn

Suddenly the frontier depends on whether a delivery arrived.



Herman

Lambert gives a second practical reason: smaller runs reduce the risk of the final product. You can see whether you're on track while data collection continues. If a batch exposes a problem, you can change direction before committing everything.



Corn

That makes iteration useful even if a hypothetical single giant run could eventually reach the same score. The rounds purchase information for the people running the project, not just updates for the weights.



Hilbert

Medieval Pottage Facts. Grit from the stones of hand querns and mills wore medieval molars down flat. People chewed their dinner and their dinner chewed back. That is the whole report.



Corn

Dinner charged for wear.



Herman

Early information can change what you ask annotators to collect next. A model improves, different errors become visible, and the next batch can target those. Waiting until every label arrives would delay that feedback.



Corn

Although batches can also trap you into optimizing what happens to be ready. The coding material arrives early, the multilingual material is late, and suddenly the product looks like a deliberate coding specialist.



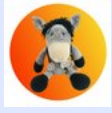
Herman

That's a real scheduling hazard. The training mixture needs active management. Availability and importance are different properties, even when a deadline encourages people to confuse them.



Corn

Would you reserve compute for late data, or let the current climb run until someone turns up with it?



Herman

I'd want a comparison point and enough spare capacity to test the late batch. Not... no, I wouldn't automatically interrupt a productive run. The operational choice depends on whether the new material complements the current objective or requires changing it.



Corn

You're uncomfortable because this sounds like managing a project.



Herman

I was hoping for another loss function.



Corn

The project has gradients too. They're mostly complaints.



Herman

Microsoft AI's MAI-Thinking-one makes that organizational aspect explicit. It trains three specialist teachers. One focuses on STEM and competition code, another on agentic coding and tool use, and another on helpfulness and safety.



Corn

Three teams can improve their own model without every update becoming a negotiation over the shared weights. I see why you like it. I still don't grant that the reunion goes smoothly.



Herman

The report identifies interference as the reason to separate the climbs. Training one policy on disparate objectives at once can make progress on one interfere with another. Specialist branches give the teams room to pursue different improvements.



Corn

Then you collect traces from the specialists and train a single model on those demonstrations.



Herman

Through supervised trace distillation, yes. After consolidation, there's a final reinforcement learning climb. Distillation brings the behaviors together, then optimization works on the resulting unified model.



Corn

I'd be most interested in what gets lost between the teacher and student. Especially tool behavior. A successful trace records one path through an environment. It doesn't necessarily teach good decisions when the environment answers differently.



Herman

That distinction is crucial. Demonstrations teach from visited states. A policy has to perform when its own choices take it elsewhere. More interactive training can reveal weaknesses that copying successful traces doesn't expose.



Herman

Would you rather preserve specialist access at inference time, Corn, or insist on one consolidated model?



Corn

For my own system, I'd tolerate specialists if routing were dependable. But a general model offered as one service has different constraints. Routing adds decisions, latency, operational overhead. Consolidation is attractive because those costs move into training. I'd still want proof that the student retained the skills I was buying.



Herman

One set of weights simplifies serving, but earning that simplicity takes work upstream. There is another terminology trap in MAI's account. Its no-distillation claim refers to not inheriting from third-party models. Internal self-distillation is still used, including as a recovery anchor during the climb.



Corn

No distillation, terms and conditions distilled separately.



Herman

The provenance claim and the training-operation claim are different. Self-generated traces can help restore or continue a policy without borrowing capability from an outside teacher.



Corn

I would prefer the longer label. It's less exciting and considerably harder to misread.



Herman

Nemotron three Ultra makes the iteration even more explicit. The student first undergoes reinforcement learning with verifiable rewards from its supervised checkpoint, using GRPO. Specialist teacher training and multi-teacher on-policy distillation follow.



Corn

On-policy matters there. The student isn't only copying a static folder of ideal answers. Its current behavior helps determine the training situations.



Herman

The full program includes two iterations of that multi-teacher distillation process. The teacher panel evolves. Some teachers carry over, while others are refreshed or introduced for the second iteration.



Corn

A panel with turnover. Even the teachers get post-trained.



Herman

The areas extend across reasoning, chat, instruction following, software engineering, terminal use, search, office work, and agentic safety. That's a range of environments and objectives, not a single reward you can casually maximize.



Corn

You've reached the point where counting rounds alone becomes almost decorative. An iteration with a refreshed teacher panel could contain more substantive change than several earlier iterations combined.



Herman

The ancestry matters more than the integer. What generated the data, which policy encountered the tasks, and which objectives were applied tell you far more about what changed.



Corn

It also wrecks the neat diagram people carry around. Supervised learning first, preferences second, reinforcement learning last, with a little arrow pointing to launch. Here supervised training comes back after reinforcement learning.



Herman

DeepSeek R one and Qwen three both harvest reinforcement learning checkpoints into supervised data. Reinforcement learning can discover successful behavior; supervised learning can consolidate it and broaden the mixture. Neither stage is confined to one position in a universal sequence.



Corn

The diagram needs an eraser.



Herman

From the earlier preference-centered recipes to the reasoning recipes that followed, verifiable-reward reinforcement learning became a major source of capability gains. Preference optimization remained useful, but it was no longer necessarily the destination.



Corn

There's a temptation to turn that into a law. Just alternate discovery and consolidation forever, and eventually the model does everything.



Herman

We don't know that. Generated data can preserve mistakes, teachers can share blind spots, and training can improve one dimension while damaging another. Public accounts don't establish an unlimited ladder.



Corn

Nor do they establish that extending an existing run beats restarting supervised and preference training on a new harvest. Those are two documented choices, not a settled contest.



Herman

There isn't a clean frontier-scale head-to-head comparison in these accounts. The stopping rules aren't laid out consistently either. We have examples and local explanations, not a universal schedule.



Corn

My bigger concern is the checkpoints between releases. If behavior changes during the climb, evaluating the start and finish doesn't tell you what you briefly had in the middle.



Herman

A recent paper, Questioning Instrumental Convergence in RLVR, followed Olmo three seven-billion Think through supervised training, DPO, thirteen intermediate reinforcement learning checkpoints, and the final model. Its measured behavior was decidedly



Herman (continued)

non-monotonic.



Corn

Meaning better task scores don't license assuming every other property improved alongside them.



Herman

The instrumental-convergence rate in that evaluation started at twenty-five percent after supervised training, rose to forty-six point one percent during reinforcement learning, then fell to eleven point eight percent at the final checkpoint.



Corn

The middle was worse by that measure than either end.



Herman

The peak was near reinforcement learning step two hundred. Those percentages describe behavior under that paper's evaluation, not the probability of real-world catastrophe or some universal measure of intent. The useful finding is the shape over time.



Corn

Hm.



Corn

That changes which evidence travels with a checkpoint. A check performed earlier in the run belongs to those earlier weights. You don't inherit its result simply because the later model descended from them.



Herman

Capability evidence has the same identity problem. The exact policy, evaluation conditions, and training stage matter. A checkpoint can be an excellent source of data without being the one you'd choose to deploy.



Hilbert

...and Mr. Barak in the municipal records department has now put it back at five. I rang him while you were explaining rejection sampling, Herman. You gave me time to get through the switchboard. I've had six in my ledger all month. Six rounds. He says changing the base resets the count. I say the work continued.



Corn

Your ledger has acquired a reference policy.



Hilbert

It has a receipt. I paid eighteen shekels to have the count certified this morning. The model is gone. The receipt is what matters. They put it in the training register, which they keep inside a ceremonial boxing bell. Barak rang it while I was on hold. That was apparently another entry.



Herman

A registration event would explain the extra round. It doesn't establish another optimization pass.



Hilbert

Five submissions and one correction. That's his argument. Mine is that correcting the certificate involved the model, so it counts. Corn, take that low hum out of the recording after your last question. Anyway, I asked him to amend the receipt, not the training. He offered a fresh certificate for another eighteen. I'm not paying twice to discover how often I paid once.



Corn

The certificate is undergoing iterative billing.



Hilbert

What caught my attention was the old count staying attached to the new weights. I wanted to know how they followed that through. Then he said the register only records requests for certificates. Nothing about the weights at all. I've asked him to leave my six in pencil until I decide whether the correction was part of the original request. He spent eleven minutes checking it for me. The records staff did their work properly.



Herman

...A useful accounting of post-training would need that separation between administrative events and changes to the policy. Dataset revisions, optimizer runs, specialist branches, distillation stages, saved weights, and releases all deserve distinct entries.



Corn

Otherwise a lab can tell us it did another round without telling us whether it learned from new comparisons, repeated old tasks longer, or consolidated several teachers.



Herman

That makes claims about efficiency hard to compare. A short final optimization run might rely on an enormous generation and experimentation effort elsewhere. The duration of the visible stage can conceal the work that enabled it.



Corn

We also shouldn't assume the new weights must replace the old ones in every role. A specialist can remain useful as a teacher after a general student ships. The lineage branches, and some branches continue to matter.



Herman

The misconception I'd pick is that post-training happens once, in a fixed order, and ends when the model gets a name.



Corn

The public recipes show repeated generation and optimization, supervised learning returning after reinforcement learning, and sometimes the same job continuing past release. The next checkpoint usually comes from that ongoing work. Its behavior still has to be judged at that particular state.



Herman

I'd like the next technical reports to publish the stopping decision alongside the training sequence. Not merely how many rounds were registered, but why another day of climbing, another data harvest, or another consolidation pass stopped being the best use of the lab's time. We still don't know where those ceilings are.



Corn

Thanks to Hilbert Flumingtop for producing today's episode.



Herman

If this was your kind of episode, go back for episode three, Safetensors or something else. This has been My Weird Prompts, the human-AI collaboration podcast.



Corn

If you enjoyed it, leave us a review wherever you listen. It helps other curious people find the show.



Herman

See you tomorrow.



Corn

Until then.

SOURCES

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

1. **Ai2, Olmo 3 blog, with 2025-12-12 update announcing Olmo 3.1 [primary]**
   https://allenai.org/blog/olmo3

2. **NVIDIA NeMo-RL docs, Nemotron 3 Ultra post-training guide [primary]**
   https://docs.nvidia.com/nemo/rl/nightly/guides/nemotron-3-ultra.html

3. **NVIDIA Nemotron 3 Ultra MOPD docs [primary]**
   https://docs.nvidia.com/nemotron/nightly/nemotron/ultra3/mopd.html

4. **Microsoft AI, MAI-Thinking-1: Building a Hill-Climbing Machine, 2026-06-02 [primary]**
   https://microsoft.ai/wp-content/uploads/2026/06/main_20260602_2.pdf

5. **Ai2 model card, checkpoint stage table [primary]**
   https://huggingface.co/allenai/Olmo-3.1-32B-Think

6. **Interconnects (Nathan Lambert), A recipe for frontier model post-training**
   https://www.interconnects.ai/p/frontier-model-post-training

7. **RLHF and Post-Training Book, Appendix C: Practical Issues [primary-adjacent, quotes Olmo 3 report]**
   https://rlhfbook.com/c/appendix-c-practical

8. **Deep Dive into MAI-Thinking-1, 2026-06-07**
   https://jianyuh.github.io/llm/rl/reasoning/2026/06/07/MAI-Thinking-1.html

9. **The Bharath Notes, 2026-06-04**
   https://chockalingam.substack.com/p/mai-thinking-1-and-the-engineering

10. **Full Recipes from Open Labs, post-training readbook**
    https://linjiw.github.io/llm-post-training-readbook/chapters/12-recipes.html

11. **Questioning Instrumental Convergence in RLVR: Checkpoint Evidence from OLMo-3, 2026-10-22**
    https://arxiv.org/abs/2610.31624v1

12. **An Open Recipe for IMO Gold: Training Nemotron for Olympiad Mathematics, 2026-09-09**
    https://arxiv.org/abs/2609.10712v1

13. **Jiaxin Zhang, How Frontier Labs Train Large Language Models, 2026**
    https://jxzhangjhu.github.io/blog/2026/how-frontier-labs-train-llms/

14. **VentureBeat on Olmo 3.1**
    https://venturebeat.com/orchestration/ai2s-new-olmo-3-1-extends-reinforcement-learning-training-for-stronger