Post-training: three disciplines in a trenchcoat

Chris von Csefalvay

Blog post published December 20, 2025 in *Chris von Csefalvay*
https://doi.org/10.59350/qg4zx-1sq02

Abstract

TipHey, I'm writing a book about this! I'm actually writing a book about this stuff. It turns out there isn't a lot of literature on how to do post-training at the level too big for single-GPU laptop-sized hobby projects and requiring enterprise reliability on one hand, but not quite at the scale of multi-team distributed post-training you'd get in foundation labs.

Keywords

Epidemiology (Subfield), AI, LLMs, Post-training

Recommended citation

von Csefalvay, C. (2025, December 20). Post-training: three disciplines in a trenchcoat. *Chris Von Csefalvay*. https://doi.org/10.59350/qg4zx-1sq02

Copyright

Copyright © Chris von Csefalvay 2025. Distributed under the terms of the Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

TipHey, I'm writing a book about this!

I'm actually writing a book about this stuff. It turns out there isn't a lot of literature on how to do post-training at the level too big for single-GPU laptop-sized hobby projects and requiring enterprise reliability on one hand, but not quite at the scale of multi-team distributed post-training you'd get in foundation labs. That's a problem, because a lot of the current value in fine-tuning applications comes exactly out of that large, crucial market. I am in the last phases of putting together the manuscript for *The Frontier Playbook*, a set of curated tactics and techniques for real world operationalisation of LLMs. Sign up for updates here.

There's a comfortable fiction in AI that "post-training" names a coherent discipline. We speak of it as though the researcher fine-tuning Llama on a 3090 and the infrastructure team planning a 500MW RLHF run are engaged in the same enterprise at different magnitudes. And yes, in a very philosophical sense, they are. It's just that that philosophical acuity reflects pretty much none of the parts of reality that matter. And that's a problem for our allegedly unitary discipline.

Pre-training never had to confront this problem. The barriers to entry were so astronomical that there was never any pretence of an on-ramp. You either had foundation lab resources or you didn't, and if you didn't, you weren't doing pre-training. This hasn't really shifted – the kind of pre-training/scaling needed to create true foundation models is limited to a vanishingly small part of the world (more people have been to outer space than have managed a capital foundation model). The conversation is contained, the community is small and everyone who needs to know what something means already knows it.

Post-training is different. It is democratised, and will only become more so as efficient (esp. *memory*-efficient) methods proliferate. The barriers of entry have never been lower. At the moment, you can run toy examples of cutting edge techniques for free with the Unsloth notebooks on Google Colab. Consumer GPUs can, with some ingenuity, run fine-tuning experiments that would have been unthinkable a few years ago. The on-ramps are everywhere.

But not to everything.

Post-training is not one discipline. It is three,[1] awkwardly sharing a name. The binding constraints, the core competencies, the failure modes differ qualitatively across regimes. Perhaps most poignantly, the governing constraints are so different, they do not share a continuum.

[1] At least. Arguably there's token-scale, where users buy API access or other consumer products. There's not a lot of what would merit being referred to as post-training there, though, so we shall omit mention.

Table 1: The three regimes of post-training, distinguished by their binding constraints.

	Memscale	**Flopscale**	**Powerscale**
Binding constraint	VRAM	Unit economics	Energy
Typical scale	1 GPU (16-80 GB)	Dozens to hundreds of GPUs	Several thousands of GPUs
Team size	Solo		

https://doi.org/10.59350/qg4zx-1sq02

	Memscale	Flopscale	Powerscale
		Solo to small team of generalists	Professionally project-managed teams for various stages
Cost scale, order of magnitude $USD/ run	$10-$100s	$1,000s	$1,000,000s

There's something magical about these three realms coexisting at all. Perhaps for the first time in history, our most advanced frontier technologies are also accessible to essentially anyone who can set up Hugging Face jobs or a Colab notebook. No matter how much these regimes differ, it's remarkable that what little continuity exists, exists at all. But that continuity is quite thin. As someone who essentially lives and breathes post-training, I have seen all three of these first-hand, although I spend almost all my time towards the middle to top end of the flopscale regime. I'm a tolerated guest in powerscale land, and a weekend adventurer in memscale. All I can say is that I never had to talk to a power company about when I can throw the on-switch.

Memscale: The tyranny of VRAM

At memscale, the fundamental question is: can I fit this into memory? This is the world of the single GPU, from a 3090 through an RTX 6000, perhaps stretching to a GB200 in a DGX Spark if you're fortunate. The binding constraint is physical and absolute. Either the model, its gradients, its optimiser states and its activations fit, or they don't. There is no negotiating with VRAM.

The entire memscale toolkit exists to cope with this constraint. LoRA and its descendants (QLoRA, DoRA, the ever-expanding alphabet) represent a fundamental insight: if you cannot fit the whole thing, fit an efficient delta. Gradient checkpointing trades compute for memory. Quantisation compresses representations. Tools like Unsloth and Axolotl have packaged these techniques into something approaching accessibility for the sufficiently obsessed.

But here's what makes memscale genuinely different as a regime: scaling is lumpy. If you run out of memory, your options are to optimise harder or to acquire another GPU, assuming you still can in the current market. There is no smooth gradient of resources. You're either within your envelope or you're not, and if you're not, you hit a wall that you can generally supervene at the cost of doubling your entire GPU investment. The memscale practitioner's core competency is OOM avoidance. Everything else is secondary.

This creates a particular kind of craft knowledge, somewhat akin to the art of the demoscene coder who could put entire animated demos into less than a modern social media avatar takes up in space. The memscale expert knows exactly which layers to freeze, which LoRA rank to choose for a given VRAM budget, how to schedule batch sizes against memory headroom, when to checkpoint and when to materialise. It is artisanal work, intimate and particular. And almost none of it transfers upward – which is why relatively few foundation lab researchers, never mind powerscale infrastructure teams, spend much time thinking about memscale techniques.

Flopscale: The discipline of unit economics

At flopscale, memory ceases to be the binding constraint in any individual sense. You have a cluster. Perhaps dozens of GPUs, perhaps hundreds. You can shard, you can parallelise, you can distribute.

What you cannot do is waste operations, because what you *also* have, typically, is a boss. This is firmly enterprise land, and enterprises care about ROI. Those TFLOPs your rack of H200s produce? They're the `I` in that abbreviation, and you best be producing some `R` on them.

The question then becomes not "can I fit it?" but "can I justify it?" The optimisation target shifts from memory efficiency to unit economics. The scarcest resource is not VRAM but organisational patience for uncertain outcomes.

Scaling here is smoother than at memscale. You don't hit walls; you encounter gradients of cost. Need more compute? Provision more nodes. The constraint is fiscal, not physical. But this smoothness is double-edged. At memscale, the walls enforce discipline. At flopscale, you have enough rope to hang yourself, and thus the flopscale practitioner is expected to bring a level of judgment about what to run when that memscale hobbyists don't have to worry about by and large and powerscale teams… well, to most, it's a rounding error.

This is a rather critical realm, and yet, paradoxically, perhaps the most woefully underserved. There are clear risks and failure modes to flopscale work, and you are not going to be able to teach yourself how to run a model on a cluster of 64 H200s from weekend tinkering.[2] You need to understand distributed systems, job scheduling, data pipeline reliability, cost monitoring, failure recovery and a host of other concerns that don't arise at memscale. Yet there are precious few resources targeted at this regime specifically. Most of the literature is either memscale hobbyist-focused or powerscale foundation lab-focused (although at the moment, regrettably, the industry is going through its mediaeval guild phase, where the secrets of building cathedrals are jealously guarded and only passed on from master to apprentice after the latter has proven itself worthy). Flopscale is the neglected middle child of post-training – the realm where most real-world applications will be built, and yet the one with the thinnest knowledge base.

[2] Please don't take a HELOC to fund your first 120B full finetune on a bunch of A100s.

Recipe: Cassoulet (sort of)

- 500g dried white beans, soaked overnight
- Duck confit (4 legs, or make your own if you have the week)
- 400g Toulouse sausage
- 200g salt pork or pancetta
- Onion, carrot, celery, garlic
- Tomato paste, bay leaves, thyme
- Good stock, ideally duck

https://doi.org/10.59350/qg4zx-1sq02

Brown meats separately. Build a soffritto. Layer everything in a Dutch oven with the beans. Add stock to cover. Bake at 150°C (that's 300°F in freedom units) for three hours minimum, breaking the crust that forms and pushing it down every hour. The crust reforms. You break it again. This is the cassoulet ritual.

Worth it? Depends on your objective function. ***

This is, I think, the hardest regime to operate in well. The memscale practitioner has constraints that enforce focus. The powerscale practitioner has resources that permit exploration. The flopscale practitioner has neither the discipline of poverty nor the freedom of abundance. They must construct their own discipline, and they must do so while delivering demonstrable business value.

Powerscale: The infrastructure of nations

At powerscale, we measure not in gigabytes or teraflops but in gigawatts. The binding constraint is raw energy: what you can power, cool and sustain. This is foundation lab territory, the handful of organisations that can credibly discuss training runs straining national electrical infrastructure.

I won't dwell long here, partly because anyone who is habitually operating at this scale is already deeply familiar with the peculiarities of their narrow place in it. This is 'big science'. I am reminded of Richard Rhodes's description of the Manhattan Project as a massive industrial enterprise. Sure, I thought, I get the first part, but… industrial? We think of it as an achievement of science – right up until we realise that just keeping Hanford running took more people per shift than the entire project had physicists. And powerscale AI engineering is no different. For every AI researcher, there are fleets of infrastructure engineers, data centre ops, civil engineers, logistics coordinators and myriad others ensuring that the lights stay on and the gradients keep flowing. I have trained some absurdly large models, but I don't routinely have to consider geopolitics or the national grid. Neither do powerscale practitioners, but they do have to deal with people who do.

The diffusion problem

This brings us to the crux of the matter. Techniques do diffuse across scales, but the diffusion is slow, uneven and often lossy. If there is such a thing as a post-training discipline, it is largely focused on solving the same vague problem in world so different that the solutions themselves need an effort to be feasibly translated, if at all possible to do so.

Consider recent work like Khatri, Madaan et al.'s ScaleRL paper. It would be hand-wavey to claim that such techniques apply only to the narrow powerscale slice. Clearly, insights about reinforcement learning at scale have downstream applicability. But the applicability is not automatic. The flopscale practitioner cannot simply import powerscale techniques wholesale. They must translate, adapt, figure out what survives the transition and what doesn't.

Tools like Unsloth accelerate this filtration significantly. They package techniques that originated in higher-resource environments into forms usable at memscale. This is genuine democratisation. But there will always be techniques that don't translate, approaches that only make sense given

certain resource assumptions, optimisations that are pointless below (or above) certain thresholds.

If you work in post-training, your first task is to know which game you're playing. The memscale practitioner who imports flopscale assumptions will waste time on approaches they cannot execute. The flopscale practitioner who ignores powerscale developments will miss techniques that could, with adaptation, provide advantage. The flopscale practitioner who imports memscale habits will under-utilise available resources.

The hardest position, I think, is flopscale. You have enough resources to attempt almost anything but not enough to attempt everything. You face genuine choices about which techniques to adopt, which to adapt and which to ignore. You must show ROI while navigating a tactical landscape that wasn't designed with your constraints in mind, and remain competitive with players who operate much larger resource envelopes for less focal problems – or know how to "get it right on a 3090" but are blissfully unaware of just how much that is not going to be acceptable in an actual enterprise deployment.

This is not a solved problem. We lack good heuristics for technique translation across scales. We lack systematic understanding of what survives the transitions and what doesn't. We lack, frankly, recognition that the problem exists.

Perhaps the first step is simply to stop pretending that "post-training" names a unified discipline. It doesn't. We have three disciplines awkwardly sharing a name, and the sooner we recognise this, the sooner we can develop the differentiated knowledge each regime demands while maintaining the channels to traffic insights between them.

https://doi.org/10.59350/qg4zx-1sq02