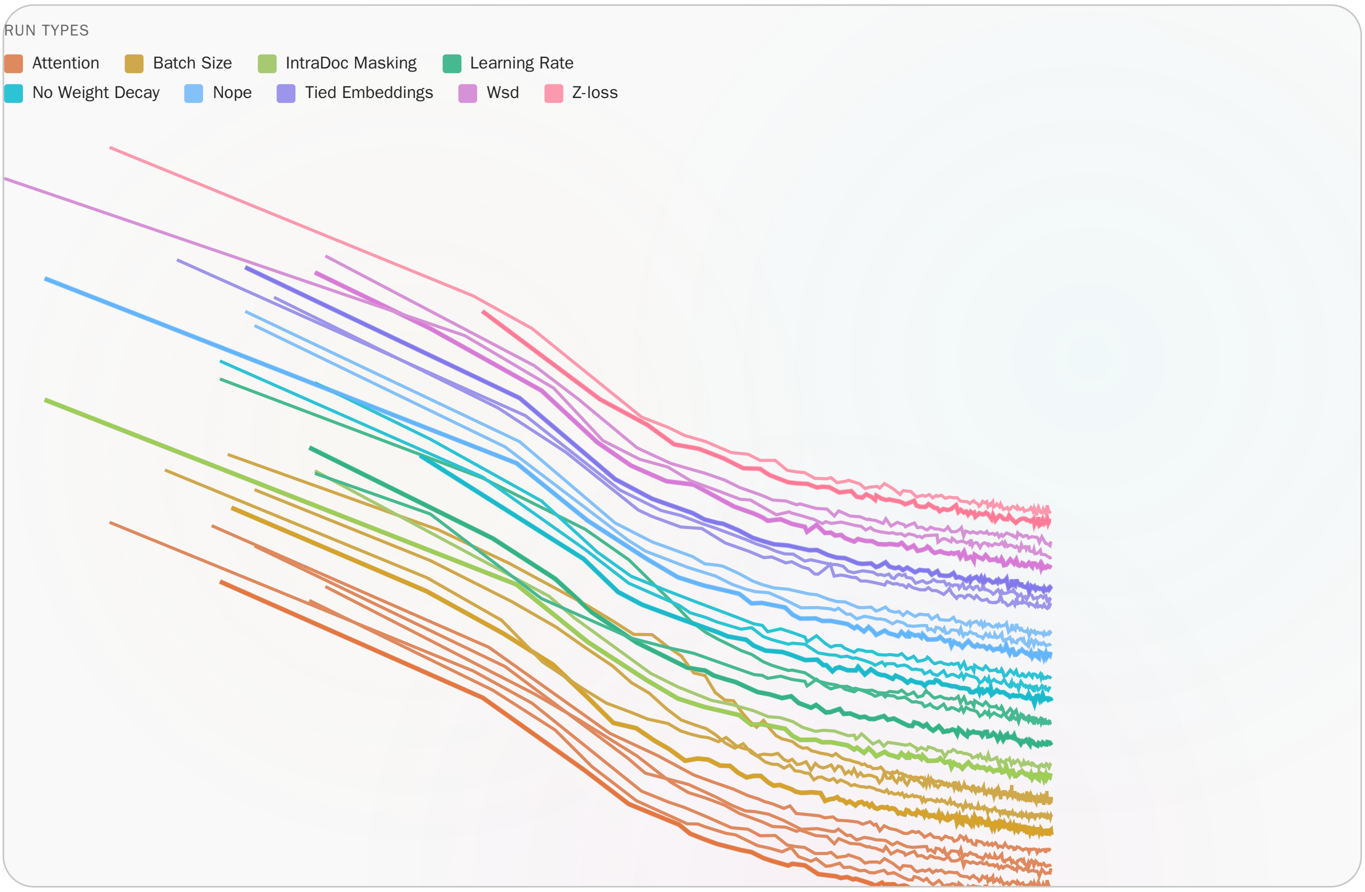
The Smol Training Playbook:
The Secrets to Building World-Class LLMs



A practical journey through the challenges, decisions, and messy reality behind training state-of-the-art language models

AUTHORS

Loubna Ben Allal, Lewis Tunstall, Nouamane Tazi, Elie Bakouch, Ed Beeching, Carlos Miguel Patiño, Clémentine Fourrier, Thibaud Frere, Anton Lozhkov, Colin Raffel, Leandro von Werra, Thomas Wolf

AFFILIATION

Hugging Face

PUBLISHED

Oct. 30, 2025

Introduction

What does it actually take to train a high-performance LLM today?

Published research makes it look straightforward: strategic architecture choices, carefully curated datasets, and sufficient compute. The results are polished, the ablations are structured and clean. Every decision seems obvious in hindsight. But those reports only show what worked and apply a bit of rosy retrospection – they don't capture the 2am dataloader debugging sessions, the loss spikes, or the subtle tensor parallelism bug (see later!) that quietly sabotages your training. The reality is messier, more iterative, and full of decisions that don't make it into the final paper.

Join us as we look behind the scenes of training SmolLM3, a 3B multilingual reasoning model trained on 11T tokens. This is not an ordinary blog post, but rather the untangling of a spiderweb of decisions, discoveries, and dead ends that led to deep insights into what it takes to build world-class language models.

It is also the final opus in our model-training long-form series: we've worked through building datasets at scale (FineWeb), orchestrating thousands of GPUs to sing in unison (Ultra Scale Playbook), and selecting the best evaluations at each step of the process (Evaluation Guidebook). Now we shape it all together to build a strong AI model. We'll walk you through the complete journey – not just the final recipe that worked, but the failures, infrastructure breakdowns, and debugging processes that shaped every decision.

The story reads like a drama: you'll see how promising small-scale ablations sometimes don't translate at scale, why we restarted a training after 1T tokens, how we balanced the competing objectives of multilinguality, math, and code while maintaining strong English performance, and finally how we post-trained a hybrid reasoning model.

We also tried to avoid a cold list of all we did in favour of an organized story through our adventure. Think of this as a guide for anyone trying to go from "we have a great dataset and GPUs" to "we built a really strong model". We hope being this open will help close the gap between research and production, and make your next training run a little less chaotic.

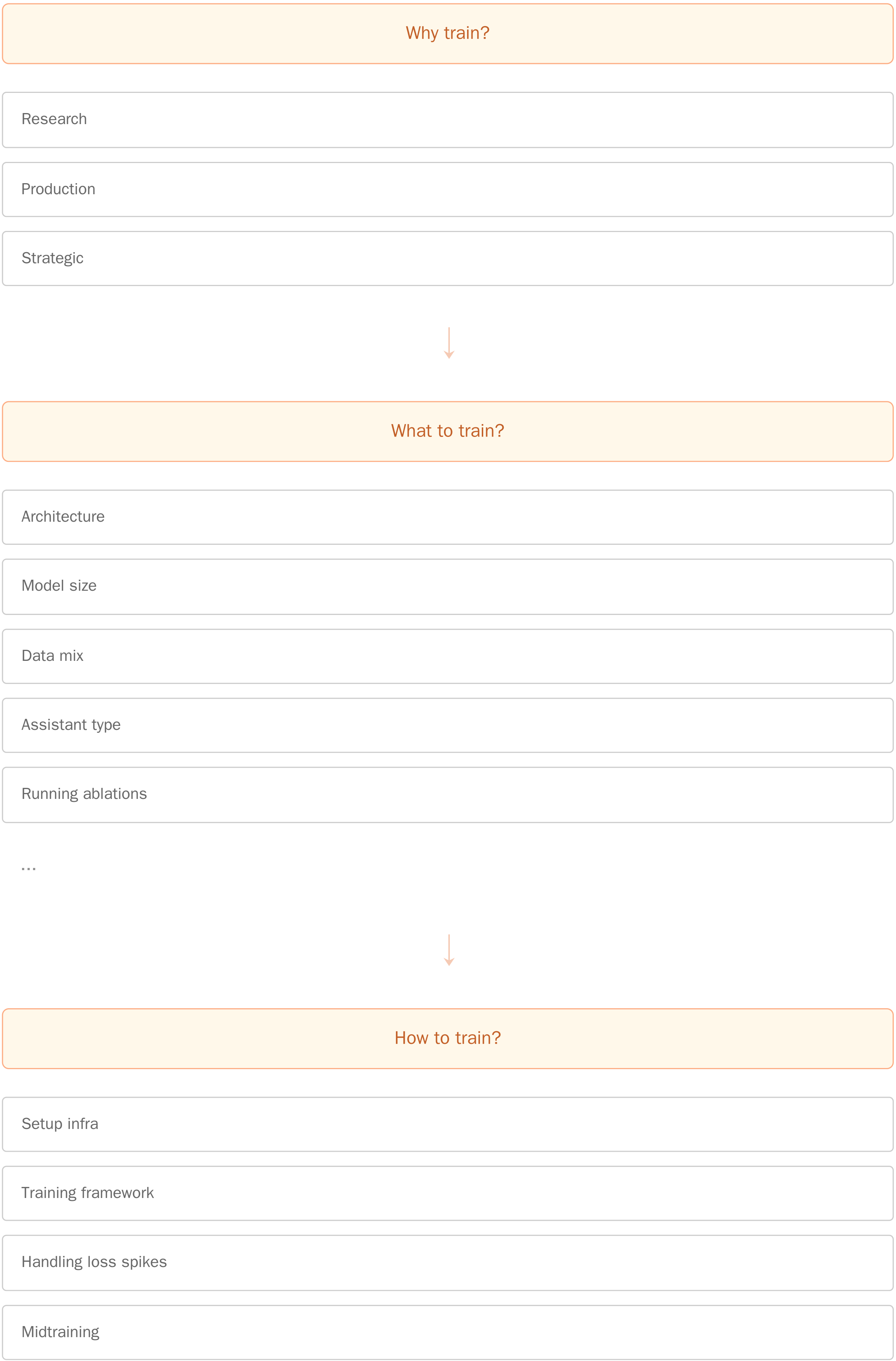
How to read this blog post

You don't need to read this blog post from top to bottom, and at this point it's too long to realistically read end-to-end in one sitting anyway. The blog post is structured in several distinct pieces that can be skipped or read individually:

- Training compass: A high-level discussion about whether or not you should pretrain your own model. We walk you through fundamental questions to ask yourself before burning through all your VC money, and how to think systematically through the decision process. This is a high-level section, if you want to skip straight to the technical content, scroll quickly past this part.
- Pretraining: The sections following the training compass cover everything you need to know to build a solid recipe for your own pretraining run: how to run ablations, select evaluations, mix data sources, make architecture choices, tune hyperparameters, and finally endure the training marathon. This section also applies if you're not planning to pretrain from scratch but are interested in continued pretraining (aka mid-training).
- Post-training: In this part of the blog you'll learn all the tricks needed to get most out of your pretrained models. Learn the whole post-training alphabet starting with SFT, DPO and GRPO as well as the dark arts and alchemy of model merging. Most of the knowledge about making these algorithms work well is learned through painful lessons, and we'll share our experience here to hopefully spare you some of them.
- Infrastructure: If pretraining is the cake and post-training is the icing and cherry on top, then infrastructure is the industrial-grade oven. Without it, nothing happens, and if it's broken, your happy Sunday baking session turns into a fire hazard. Knowledge about how to understand, analyse, and debug GPU clusters is scattered across the internet in various libraries, docs, and forums. This section walks through GPU layout, communication patterns between CPU/GPU/nodes/storage, and how to identify and overcome bottlenecks.

So where do we even start? Pick the section that you find most exciting and let's go!

Training compass: why → what → how



Why train?

- Research
- Production
- Strategic

What to train?

- Architecture
- Model size
- Data mix
- Assistant type
- Running ablations

...

How to train?

- Setup infra
- Training framework
- Handling loss spikes
- Midtraining