Qwen3.5-9B LoRA fine-tuning on Apple's WWDC26 recipe: pipeline parallel over four MacBook Pro M5 Max with NukeTorch

Klay Moon, NukeTorch · Report v1.2 · 4 Apple M5 Max, Thunderbolt bridge

Headline

- **5,358.5 trained tokens per second**, pipeline parallel over four Apple M5 Max (NukeTorch's Choo-Choo Relay Train), over the two rested runs of three default-configuration runs (5,336.1 and 5,381.1; the third, started heat-soaked, ran at 5,068.6, Table 2): **1.02×** our data parallel on the same four Macs, recipe and 1,462,055 tokens. Data parallel ran twice, 5,230.6 and 5,128.6 (2.0% apart); we compare with the faster and read the two methods as equal, with pipeline parallel no slower.
- **Comparable final loss:** validation / test loss 1.575 / 2.154 against data parallel's 1.627 / 2.245 (Table 4). The three default-configuration runs end with byte-identical adapters.
- **Energy: 95.26 mJ per trained token** against data parallel's 102.77: 7.3% less (one run each; Table 5).
- **Over 8× Apple's WWDC26 run:** Apple presented this recipe on four M3 Ultra Mac Studios at about 600 tokens per second, its spoken figure (Table 3). Every NukeTorch run here, pipeline or data parallel, is more than 8× that figure; the slowest, the heat-soaked run 3, is 8.45× and the headline rate is 8.93×.

Table 1. Speed (1,000 steps, 1,462,055 trained tokens per run)

Source	Machines	Steps time (s)	Trained tokens/s	Against data parallel
NukeTorch, pipeline parallel (Choo-Choo Relay Train), runs 1 + 2	**4 Apple M5 Max**	**545.7 (both runs)**	**5,358.5**	**1.02×**
NukeTorch, pipeline parallel, run 1	4 Apple M5 Max	274.0	5,336.1	1.02×
NukeTorch, pipeline parallel, run 2	4 Apple M5 Max	271.7	5,381.1	1.03×
NukeTorch, data parallel	4 Apple M5 Max	279.5	5,230.6	1.00×

Trained tokens per second = the run's trained tokens (padding excluded) ÷ the sum of its step times, step 0 included; runs 1 + 2 = both runs' tokens ÷ both runs' times.

Table 2. Other pipeline-parallel runs of the same recipe (disclosed)

Run	Steps time (s)	Trained tokens/s	Against data parallel
run 3: the default configuration, started heat-soaked	288.5	5,068.6	0.97×
run 4: an earlier pipeline configuration	279.1	5,238.1	1.00×

Run 3 is the default configuration started heat-soaked after back-to-back runs: the Macs' clock speeds fell after the first minute, and its rate per 100 steps fell with them. It ends with the same adapter bytes as runs 1 and 2. Run 4 is an earlier configuration of the pipeline; its final validation / test loss is 1.523 / 2.145.

Table 3. Published and as shipped

Source	Machines	Form	Run length	Trained tokens/s
Apple, as spoken (WWDC26)	4 M3 Ultra Mac Studios	data parallel (MLX)	not stated	~600
MLX as shipped, four Macs	4 Apple M5 Max	data parallel	—	not run
MLX as shipped, four Macs	—	pipeline parallel	—	no maker path

Apple's figure is a spoken approximation from its session, on its machines: "Single M3 Ultra is processing around 180 tokens per second, while on the cluster we process around 600 tokens per second" (WWDC26 session 233, "Explore distributed inference and training with MLX", chapter "Distributed fine-tuning" (figure at 18:10)). Its four-Mac command, as shown on the session's page, is the recipe here: `mlx.launch --hostfile "hostfile.json" -- mlx_lm.lora --model "Qwen/Qwen3.5-9B" --data "mlx-community/wikisql" --train --batch-size 16`. The machines differ, four M3 Ultra Mac Studios against four MacBook Pro M5 Max here; the ratio compares the same recipe as each side ran it, against Apple's own figure. MLX as shipped was not run on four Macs. In an earlier run of MLX as shipped on one M5 Max (MLX 0.32.4 / mlx-lm 0.32.2, Apple's one-Mac command with `--batch-size 4`, the per-Mac share of this recipe), it stopped with "Insufficient Memory" before its first step. MLX ships no pipeline-parallel training path for this recipe.

Table 4. Comparable final loss

Measure	NukeTorch, pipeline parallel	NukeTorch, data parallel	Difference
Validation loss (100 rows)	1.575	1.627	-3.2%
Test loss (100 rows)	2.154	2.245	-4.0%

Final losses of the adapters after 1,000 steps on the recipe's 100 validation and 100 test rows; the other data-parallel run ends at 1.625 / 2.223. The recipe overfits: in both methods validation loss is lowest near step 200 (data parallel 0.994; pipeline parallel, measured on run 4, 0.975 at step 200) and rises after it, so the lower final validation loss reflects slower fitting, not a better model. Training loss, mean of each 100 steps:

Steps	Pipeline parallel (runs 1 and 2, identical)	Data parallel	Difference
1–100	1.1234	1.1074	+1.5%
101–200	0.6964	0.6859	+1.5%
201–300	0.5105	0.5022	+1.6%
301–400	0.3852	0.3731	+3.2%
401–500	0.2822	0.2770	+1.9%
501–600	0.1906	0.1840	+3.6%
601–700	0.1425	0.1407	+1.3%
701–800	0.1168	0.1165	+0.3%
801–900	0.1072	0.1046	+2.5%
901–1,000	0.1020	0.1007	+1.2%

Table 5. Energy (whole machines, four Macs)

Source	Energy for 1,462,055 tokens (Wh)	mJ per trained token
NukeTorch, pipeline parallel, run 2	**38.686**	**95.26**
NukeTorch, data parallel	41.737	102.77

Pipeline parallel used 7.3% less energy per trained token in these two runs (one run each; no error bar). Meter: each Mac's battery-controller load counter, the whole machine, averaged between its refreshes (about once a minute) inside the timed steps; no Mac's battery was charging in either run. Run 2 started after 22 minutes of rest; the data-parallel run's rest before its start is not recorded. Run 1 is not compared: a battery charged during it (on 1 of the four Macs). MLX energy was not measured.

Disclosures

- **Two default runs, two builds.** Run 2 used a newer build of the engine than run 1; both end with byte-identical adapters and the same loss at every step.
- **Run-to-run variation.** The same configuration ran 6.2% faster rested (run 2) than heat-soaked (run 3); the two data-parallel runs differ by 2.0%; pipeline and data parallel differ by 2.4%.
- **Same batches.** Every pipeline run and the data-parallel runs train the same micro-batches in the same order, checked byte for byte (4,000 per run).
- **Loss path.** Pipeline-parallel and data-parallel training follow different loss paths step by step (Table 4) and end at the same place.

Method

Setting	Value
Recipe	Apple's WWDC26 four-Mac LoRA recipe: Qwen/Qwen3.5-9B, mlx-community/wikisql, global batch 16 (4 per Mac), LoRA rank 8, scale 20, layers 16–31, Adam at 1e-5, 1,000 steps
Tokens	1,462,055 trained tokens per run (padding excluded), the same batches in the same order in every run
Method	NukeTorch pipeline parallel over four Macs (the Choo-Choo Relay Train); NukeTorch data parallel over the same four Macs
Rate	Trained tokens ÷ the sum of step times, step 0 included
Builds	pipeline parallel: commits 68590d81 (run 1) and dc8856b5 (run 2), clean trees; data parallel: 02c4b327, clean
Machines	Four MacBook Pro, Apple M5 Max, joined by a Thunderbolt bridge; mains power; a process check on every Mac before each run
Not measured	Peak memory; MLX on four Macs (not run); MLX energy

How to reproduce

1. **Machines:** four MacBook Pro 16-inch, Apple M5 Max (18-core CPU, 40-core GPU, 48 GB), joined by a Thunderbolt bridge, on mains power.

2. **Cooling:** No external fans; each Mac's own fans under automatic control (declared by NukeTorch). Our other published benchmarks used external fans; this one did not.
3. **Model and data:** Qwen/Qwen3.5-9B at revision c20223623576; mlx-community/wikisql at revision 886acf6d49be (1,000 training, 100 validation and 100 test rows).
4. **Recipe:** Apple's four-Mac command from its session (Table 3), unchanged: global batch 16 (4 per Mac), LoRA rank 8, scale 20, layers 16–31, Adam at 1e-5, 1,000 steps.
5. **Engine:** NukeTorch pipeline parallel over the four Macs, builds as listed under Method (available to licensees); the same micro-batches in the same order in every run.
6. **Before each run:** at least ten minutes at rest, then a process check on every Mac. Run 3 started without the rest and is reported as heat-soaked.
7. **Measure:** trained tokens ÷ the sum of step times, step 0 included (Table 1).

Prepared by NukeTorch Inc. Report v1.2. Author-run measurements; no independent reproduction yet.