Verified Self-Improvement and Test-Time Scaling on One Consumer GPU: A Reproduction Study with Confirmations and a Substitution Effect

Anonymous authors
Paper under double-blind review

Abstract

Small, locally hosted language models remain unreliable on multi-step reasoning tasks. Most reported improvements: verified self-training, test-time scaling (TTS), and confidence-based routing, have been evaluated on full-precision models running on server-class hardware. We revisit these claims under realistic consumer constraints: a single *4-bit* 9B model running on *one* consumer GPU (RTX 5080, 16 GB). We reproduce four published results from Think–Prune–Train, self-consistency, s1 budget forcing, and on-device routing. We also perform an additional comparison not studied in Think–Prune–Train: self-training versus frontier-teacher distillation. All reported gains replicate on consumer hardware. We then report two findings. **N1**: Fine-tuning and test-time scaling act as substitutes. A weak base model (81.4% sampled maj@1; ≈80% greedy) and a self-trained model (86.4%) both converge to around 94.0% under maj@16. Additional training improves low-compute performance, but its advantage disappears as test-time compute increases. The same pattern holds on progressively harder benchmarks, from GSM8K to MATH-500 and AIME. **N2**: a GSM-8K comparison of self-training vs frontier-teacher distillation with cross-family GPT-5.5 (89.5%) and same family Qwen3.5-27B (90.0%) and Qwen3.5-397B (92.5%) models. We release all code used in the study in an anonymized repository.

1 Introduction

Frontier models answer grade-school math (GSM8K) at 95–99%. A 4-bit 9B model fine-tuned on consumer hardware tops out near 88–90%. The methods proposed to close this reliability gap for small models (verified self-training (Costello et al., 2025), parallel test-time scaling via self-consistency (Wang et al., 2023), sequential test-time scaling via budget forcing (Muennighoff et al., 2025), and on-device routing (Saad-Falcon et al., 2026)) were each validated at full precision on server-class hardware. The *consumer-GPU* setting (a single 16 GB card, 4-bit quantization) is the setting in which most practitioners would deploy a local model, and the setting in which these published claims have *not* been systematically tested.

We test the four published claims (C1-C4), and report two of our own extensions, summarized in Table 1. Each section of the paper states a published claim, then *confirms*, *partially confirms*, or *fails* it in the single-GPU/4-bit setting. We deliberately reuse one verifier, one few-shot recipe, and one held-out split throughout so that every number is comparable across the training and inference axes.

Contributions. **(1)** A substitution finding (N1): at sufficient sampling budget, a fine-tune's single-shot edge is fully absorbed by self-consistency. **(2)** Self-training v/s Frontier-distillation comparison (N2): Self training gets close to frontier model distillation. **(3)** Confirmation that the reproduced claims (C1-C4) hold at 4-bit on 16 GB. **(4)** Open-source code with reproduction script and seed.

Table 1: Claims under test and their verdicts in the 16 GB / 4-bit setting. C1-C4 are reproductions of published claims; N1–N2 are new findings. **N2 is our own frontier-teacher comparison** (the original TPT paper runs *no* teacher distillation), included alongside the reproductions for context.

#	Source	Claim we test	Verdict
C1	TPT (Costello et al., 2025)	Verified self-training lifts a small model's GSM8K accuracy ∼8 pts	**Confirmed** (80→88.5%)
C2	Self-consistency (Wang et al., 2023)	maj@k lifts accuracy over greedy decoding	**Confirmed** (81→94% maj@16)
C3	s1 (Muennighoff et al., 2025)	Sequential "Wait" extension scales accuracy	**Partial** (n=30 pilot; numbers withheld)
C4	Routing (Saad-Falcon et al., 2026)	On-device routing ≈ cloud accuracy at a fraction of cost	**Confirmed in kind** (95.5% @ 2% escalation)
N1	*(new)*	Fine-tuning and TTS **substitute**, not compound	base & TPT both → 94.0% maj@16
N2	*(new)*	Self-training ≈ frontier-teacher distillation	**Yes** (88.5 vs 89.5 - *GPT5.5 distillation*)

2 Scope and Methodology

Model and hardware. All local inference and fine-tuning use Qwen3.5-9B loaded in 4-bit (Unsloth (Unsloth AI, 2024)) with QLoRA adapters (Dettmers et al., 2023; Hu et al., 2022), on a single RTX 5080 (16 GB, Blackwell `sm_120`, CUDA 12.8). The 16 GB VRAM was the limiting factor. Merged 16-bit fine-tunes do not fit a high-throughput server such as vLLM on this card, so bulk sampling stays on the 4-bit Unsloth path and larger teachers are reached only by API.

Two evaluation protocols that must not be cross-compared. We keep two protocols strictly separate. **(i)** Fine-tuned Qwen models are scored with `lm-evaluation-harness` (Gao et al., 2024), 5-shot, reporting `flexible-extract` (robust last-number match) as the fair metric because we noticed small formatting issues penalizing performance without it. **(ii)** Frontier API models are scored zero-shot with a verifier (solve, then check the parsed answer with Math-Verify) because these models don't need the few-shot harness. The API results are a rough ceiling that compares our results with the frontier, not as a head-to-head comparison.

Shared verifier. A single scorer, `gsm8k_verify` (Math-Verify (Hugging Face, 2025) with a numeric fallback), is used for the prune step of self-training, for self-consistency voting, and for the API ceiling. For the harder benchmarks (§5) the same verifier is extended to parse `$...$`-wrapped LaTeX.

Benchmarks. The primary benchmark is GSM8K (Cobbe et al., 2021) test, first n=200 problems for parallel/cascade and distillation, n=30 for budget forcing (per-problem greedy decoding is slow). The generalization study (§5) adds MATH-500 (Lightman et al., 2023; Hendrycks et al., 2021) (n=100) and AIME 2024/2025 (n=30 each).

Deviations from the original papers, stated up front as threats to validity. We diverge from the source works in four ways, each of which could shift absolute numbers: (a) 4-bit quantization vs. full precision; (b) small n (200/100/30) and a *single* seed (3407), so confidence intervals are wide (≈ ±5% at n=200) and few-point differences are noise; (c) GSM8K as the primary benchmark, where the base model is already strong; (d) a format-tuned few-shot prompt for local models vs. a zero-shot chat prompt for the API ceiling. These are revisited in §7.

Deltas vs. the official released code. Both reproduced methods have public author code (`ScalingIntelligence/TPT` (Scaling Intelligence Lab, 2025), `simplescaling/s1` (simplescaling, 2025)); we

Table 2: Deltas vs. the official released code. "Faithful" = matches the released implementation; "adaptation" = a single-GPU compromise that should not change conclusions; "deviation" = a substantive difference a reader must know about.

component	original (official code)	this study	label
TPT: training	full SFT, lr 1e-6/1e-5, cosine	4-bit QLoRA, lr 1e-4, linear	adaptation
TPT: pruning signal	exact ground-truth `==` match	Math-Verify	adaptation (more lenient)
TPT: traces kept/problem	≤ 1	≤ 2	minor
TPT: samples K	free CLI arg (defaults 2/5/10/50)	16	**faithful** (free in both)
TPT: eval protocol	temp-0.7 *sampled* Pass@1, zero-shot template, curated 500	lm-eval **5-shot greedy**, n=200	**deviation** (§3.1)
TPT: frontier-teacher distill	**none** in paper or code	our GPT-5.5 / Qwen-27B comparison	**our contribution**
s1: curtail + decode	cap thinking, force-close, greedy	same	**faithful**
s1: extend / knob	fixed swept `thinking_n_ignore`$\in \{1, 2, 4, 6\}$	opportunistic "Wait" (cap 6) + sweep the think token budget B	**deviation** (§4.4)

Input: model M; problems $\{(q_i, a_i)\}$; samples/problem K; temperature T; verifier V.
Output: fine-tuned M'.
$\mathcal{D} \leftarrow \emptyset$
for each problem (q_i, a_i):
  Think: draw K completions $c_i^{(1)}, \ldots, c_i^{(K)} \sim M(\cdot \mid q_i, T)$
  Prune: **for** each completion c: **if** $V(\text{parse}(c), a_i)$ is correct **then** add (q_i, c) to $\mathcal{D}$ (keep verified-correct only)
deduplicate and cap $\mathcal{D}$ at ≤ 2 traces per problem
Train: $M' \leftarrow$ QLoRA-SFT$(M, \mathcal{D})$
return M' (generator for the next round)

Algorithm 1: Think–Prune–Train (one round)

read both and tabulate every adaptation (Table 2), labeled faithful / adaptation / deviation, so a reviewer diffing against them sees exactly where we depart. Two things are worth noting. First, the original TPT code contains *no* teacher-distillation pipeline of any kind (its only "distill" arm trains a smaller model on the 9B's *self-generated* data), which is why our frontier-teacher comparison (N2) is an extension and not a reproduction. Second, the sample count K is a free hyperparameter in both code bases, so our K=16 is *not* a deviation.

3 Reproducing the Training Axis (C1, N2)

The training axis spends compute *before* inference, to improve the single-shot reliability of a fixed-size model. We reproduce Think–Prune–Train (TPT), add our frontier-teacher distillation comparison (N2).

The method. TPT (Costello et al., 2025) is verified self-training. The model samples its own solutions to training problems, an objective verifier keeps only the correct traces, the model is fine-tuned on those, and the loop repeats (Algorithm 1). The model gains capability by putting more probability mass on its *own* verified successes, not by imitating an external teacher. The "prune" step is data filtering by ground-truth verification, not a model-emitted critique.

Table 3: Training-axis reproduction on GSM8K (lm-eval, 5-shot, n=200). All fine-tunes plateau near 88–90%.

Variant	GSM8K (flexible-extract)	note
baseline	80.0%	n/a
TPT round 0 (self-train)	**88.5%**	C1: +8.5
TPT round 1 (self-train)	85.5%	C1: overfits (below round 0)
GPT-5.5 distill (cross-family)	89.5%	N2: ties (self-train)
Qwen3.5 27B distill (same-family)	90.0%	N2: ties
Qwen3.5-397B distill (same-family)	**92.5%**	N2: best, +4.0

3.1 TPT replicates (C1)

Starting from the 4-bit base, one TPT round lifts GSM8K from 80.0% to 88.5% flexible-extract (Table 3), a +8.5-point gain that reproduces the paper's ∼8-point claim on consumer hardware. The mechanism is visible in the sampling statistics: round-0 Pass@1 is ≈81.5% but Pass@16 coverage is ≈98% (490/500 train problems solved at least once). TPT converts that coverage gap into reliable single-shot accuracy. A second round leveled off (85.5%, potential overfit from round 0), so the gain is front-loaded. *Protocol caveat (vs. the original eval).* Our 88.5% is lm-eval 5-shot *greedy* flexible-extract on n=200. The original TPT paper reports Pass@1 as a *temp-0.7 sampled* metric on a curated 500-problem set with a zero-shot instruction template, so the 88.5%↔paper comparison is not like-for-like (different prompt, decoding, *and* eval set). Our own *sampled* Pass@1 (≈81.5%) is the closest analog to the paper's number (Table 2).

3.2 Self-training ≈ frontier-teacher distillation: our comparison (N2)

This comparison is *our own contribution*, not a reproduction: the original TPT paper and its released code run *no* frontier-teacher distillation (their only "distill" arm trains a smaller model on the 9B's *self-generated* data), so the paper's actual thesis is self-improvement *without* a teacher. We keep the training recipe fixed and change only *who authored the training traces.* A cross-family GPT-5.5 teacher solving the same 500 GSM8K-train problems (kept only if gold-verified, 485/500) yields 89.5% flexible-extract, statistically tied with TPT's 88.5%. Same family Qwen3.5 27B teacher yields similar but slightly better performance at 90.0%. A much larger same-family Qwen3.5-397B teacher reaches 92.5%, +4.0 over self-training. At single seed and n=200 this gain is within the noise band, so this is suggestive and not conclusive.

4 Reproducing the Inference Axis (C2-C4, N1)

The inference axis spends compute *at* test time on a fixed model: parallel breadth (self-consistency), sequential depth (budget forcing), and routing (cascade).

4.1 Self-consistency replicates (C2)

For each problem we draw K=16 samples at T=0.8. We report maj@k, the accuracy of a majority vote over k samples (what you actually get), and pass@k, the fraction of problems where at least one of the k samples is correct (the ceiling a perfect picker could reach, using the unbiased estimator of Chen et al. (2021)). Self-consistency lifts the base 9B by +12.6 points to 94.0% at maj@16 (Table 4, Fig. 1). This matches a 27B's zero-shot accuracy with *no* larger local model, and confirms C2 at 4-bit. The maj@k→pass@k gap (+4.5/+5.0) is the realizable-vs-oracle headroom a perfect verifier or router could capture (§4.4). An independent re-run on the same GPU reproduced this headline within 0.2 points (maj@16 93.8% vs. 94.0%, pass@16 coverage 98.5% exact) and confirmed the cascade router (§4.4) decides escalation purely from vote agreement with no gold-label leakage; we keep 94.0% as the headline and report 93.8% as corroboration.

Table 4: Self-consistency on GSM8K test (n=200, K=16, T=0.8). Both base and self-trained models level off at *exactly* 94.0% by maj@16 (N1). pass@16 is the oracle coverage ceiling. Pass@1 here is the *sampled* maj@1 (T=0.8), *not* a greedy decode (the genuine greedy/lm-eval base is ≈80%); it drifts ±1–2 pt run-to-run.

model	Pass@1	maj@4	maj@8	**maj@16**	pass@16 (oracle)
base	81.4%	90.0%	92.7%	**94.0%**	98.5%
tpt-round-0	86.4%	91.5%	93.3%	**94.0%**	99.0%

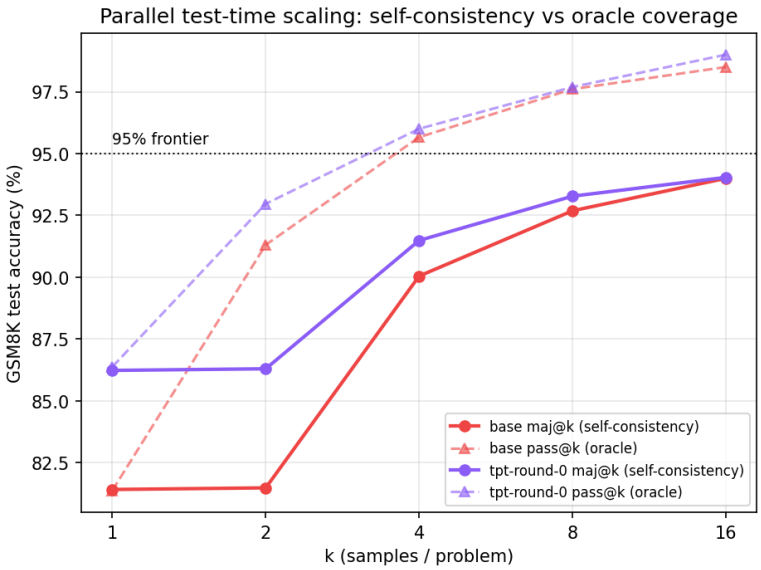

Figure 1: Parallel test-time scaling on GSM8K: maj@k (realizable) vs. pass@k (oracle coverage) for the base and self-trained 9B. Both models' maj@k curves converge to 94.0% by k=16 (the substitution effect, N1); the gap to pass@k is the headroom a verifier or router could convert.

4.2 Substitution, not compounding (N1)

The base (81.4% *sampled* maj@1) and the self-trained model (86.4%) both land close to 94.0% at maj@16 (Table 4). The TPT model's +5-point single-shot edge is fully erased once 16 samples are drawn: fine-tuning front-loads single-shot reliability, and self-consistency buys the same reliability at inference. **Fine-tuning and test-time compute substitute rather than compound.** If you intend to sample 16× at test time anyway, the fine-tune adds little *on this task*. We expect the substitution to be only *partial* where pass@k coverage is itself low, which motivates the generalization study (§5).

4.3 Budget forcing: partial replication (C3)

Sequential scaling (s1 (Muennighoff et al., 2025)) greedily decodes inside a `<think>` block: if the model closes thinking before a token budget B is spent, the close token is dropped and "Wait" is appended to force more reasoning; once B is reached, thinking is force-closed and an answer is elicited (Algorithm 2). On an n=30 base-only GSM8K pilot we sweep the think-token budget $B \in \{256, 512, 1024\}$, and accuracy rises with the budget: 43.3% at 256 tokens, 80.0% at 512, and 93.3% at 1024 (Appendix B; the per-run summary `base_budget_summary.json` is committed to the released harness). A budget that is too small cuts reasoning off early; more budget recovers and then passes the model's greedy accuracy, the sequential-scaling effect s1 reports. We mark C3 *partial* because the pilot is small (n=30, single seed) and our *extend* mechanism is not faithful to s1, not because the numbers are withheld. *Fidelity to s1.* The curtail mechanism and greedy decode are faithful to s1, but our *extend* mechanism is *not*: s1's primary knob is a fixed, swept `thinking_n_ignore`$\in \{1, 2, 4, 6\}$ (suppressing the end-of-think token a fixed number of times), whereas we

Input: model M; problem q; think-token budget B.
open the assistant turn with `<think>`
while think-tokens spent $< B$:
  greedily decode the next token
  if token is `</think>`: drop it and append "Wait" (extend: suppress end-of-thinking)
force-close `</think>` and elicit a final answer (curtail at budget B)
return parsed answer

Algorithm 2: s1 budget forcing (extend / curtail)

Input: local model M; big model B_{big}; problem q; samples k; threshold τ.
draw k samples from M and verify-vote $\rightarrow$ majority answer $\hat{a}$, plurality count p
agreement $\leftarrow p/k$
if agreement $\geq \tau$: **return** $\hat{a}$ (accept on-device; cheap, batched)
else: return $B_{\text{big}}(q)$ (escalate only the unsure tail)

Algorithm 3: Confidence-routed cascade (vote-agreement routing)

fire "Wait" opportunistically (cap `max_waits`=6) and sweep the token budget B. This mixes up s1's two separate knobs and imposes a *minimum*-budget behavior s1 lacks (Table 2). That minimum-budget forcing also over-thinks easy problems, so an adaptive budget should dominate a fixed-B schedule, which is the per-query signal the cascade exploits next.

4.4 Cascade $\approx$ on-device-routing economics (C4)

The cascade runs the self-trained 9B with self-consistency and uses its *vote agreement* (plurality count / k) as a free routing signal: if agreement $\geq \tau$ accept the local majority, else escalate the query to a larger model (Algorithm 3). Sweeping τ traces the accuracy-vs-escalation frontier (Table 5, Fig. 2). **The 9B reaches 95.5% by escalating just 2.0% of its least-confident queries to a 27B; 98% stay on-device.** This is a small gain (+1.5 over the 94.0% local maj@16, i.e. +3 of 200 problems, inside the single-seed $\pm$5% band), not a qualitative milestone, and reproduces in kind the on-device-$\approx$-cloud-at-a-fraction-of-cost claim of Saad-Falcon et al. (2026) (which itself reports coming within $\sim$3.2 pp of cloud accuracy on average). Three observations: (i) vote agreement is a strong, *free* router, with no trained verifier needed; (ii) escalation caps at 46% because 54% of problems receive *unanimous* local votes, which are almost always correct; (iii) accuracy is *non-monotonic* in τ (it peaks around 98.5% for GPT-5.5 escalation, then declines), because over-escalating confident-correct local answers to an occasionally-wrong big model slightly hurts, so there is a *compute-optimal* operating point rather than "escalate more is better."

5 Generalization: Does the Story Hold Beyond GSM8K?

The base model already scores near the top on GSM8K, so substitution (N1) might hold only on a single benchmark. The TPT model was fine-tuned *exclusively* on GSM8K-train traces, so its transfer is also open. We re-run both axes across a difficulty gradient (GSM8K, easy and in-distribution; then MATH-500, medium to hard and OOD; then AIME 2024/2025, very hard and OOD) to test the falsifiable prediction that where oracle coverage is *low*, TPT and TTS should *compound* rather than substitute.

Measurement integrity. This study makes a cross-protocol claim (a 4-bit 9B with maj@16 reaching the zero-shot frontier band on MATH-500), so we audited the extractor on real frontier output before trusting any number. On a 20-problem frontier sample, the `Final answer:`-line extractor scored 16/20, *beating* letting Math-Verify parse the whole solution (11/20, which trips on $90 \equiv 90^\circ$ and Evelyn $\equiv$ `\text{Evelyn}`); every one of the 4 misses was a genuine model error, not an extraction failure. The remaining asymmetry is one

Table 5: Compute-optimal routing on GSM8K (n=200). Local = tpt-round-0 maj@16 (94.0%); route by vote agreement. Reference big models on the same problems: Qwen3.6-27B 97.0% (mean 32.6 s/query), GPT-5.5 99.0% (3.7 s). "Oracle routing" escalates exactly the local errors.

escalate →	cheapest ≥95%	@ 6% escalated	@ 13% escalated	oracle routing
Qwen3.6-27B	**95.5% @ 2.0%**	96.5%	97.5%	98.5% @ 6%
GPT-5.5	**95.5% @ 2.0%**	96.5%	98.0% @ 18%	99.5% @ 6%

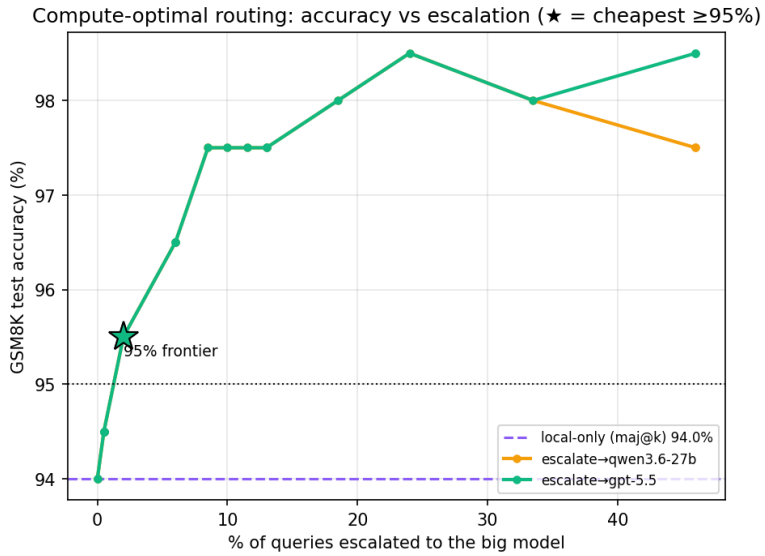

Figure 2: The cascade frontier on GSM8K: accuracy vs. fraction of queries escalated, routed by local vote agreement. The 9B reaches 95.5% by escalating only 2% of queries to a 27B (a small gain over the 94.0% local maj@16, within the single-seed noise band). Accuracy is non-monotonic in the routing threshold, so there is a compute-optimal escalation rate.

of *protocol* (format-tuned few-shot locally vs. zero-shot chat for the frontier), so frontier rows are reported as a zero-shot *reference band*, not a like-for-like ceiling.

Findings. F1: TPT transfer is real but distribution-bounded. TPT lifts Pass@1 by +5.0 on GSM8K and +10.2 on MATH-500 (60.4→70.6). It transfers OOD, and helps *more* on MATH where the base has more headroom, but collapses into noise on AIME (+0.2 on 2024, +2.8 on 2025). **F2: the substitution (N1) generalizes where coverage is high.** On MATH-500, TPT's +10.2 Pass@1 lead shrinks monotonically under voting (maj@1 +10.0 → maj@16 +1.0) and both models hit the same 95.0% coverage ceiling, moving N1 from a single-benchmark null toward a *coverage-dependent regularity*. **F3: self-consistency transfers more strongly than the fine-tune.** The maj@16-over-Pass@1 gain is larger on the harder, lower-Pass@1 task (+22.8 on MATH-500 vs. +12.5 on GSM8K). **F4: compounding is not observed; its setting is empty here.** Where coverage is low (AIME) TPT does not transfer in the first place, so there is no per-sample edge for voting to amplify (the two AIME years even disagree on the sign of the TPT effect, and n=30 is noise-dominated). **F5: the gap to frontier widens with difficulty.** The 9B with maj@16 reaches near-parity on GSM8K (94.0% vs. 96.5–99%) and meets the entire frontier zero-shot band on MATH-500 (83–84% vs. 63–81%), but 4–12% vs. ∼100% on AIME; the Olympiad tier stays out of reach regardless of test-time compute. We also ran a sequential budget-forcing pilot on MATH-500 (Appendix B): accuracy again rises with the think-token budget, and the self-trained model overtakes the base at the largest budget (90.0% vs 86.7% at B=2048).

Table 6: Local models across the difficulty gradient (single seed; n as shown). "maj gain" is maj@16 over Pass@1; "gap" is pass@16 minus maj@16. TPT's Pass@1 edge shrinks under voting wherever coverage is high (N1, OOD), and vanishes into noise where it fails to transfer. †AIME is n=30 and noisy.

benchmark	model	n	Pass@1	maj@16	pass@16	maj gain	gap
GSM8K	base	200	81.5%	94.0%	98.5%	+12.5	+4.5
	TPT (rd 0)	200	86.5%	94.0%	99.0%	+7.6	+5.0
MATH-500	base	100	60.4%	83.2%	95.0%	+22.8	+11.8
	TPT (rd 0)	100	70.6%	84.2%	95.0%	+13.6	+10.8
AIME 2024†	base	30	2.9%	8.4%	23.3%	+5.5	+14.9
	TPT (rd 0)	30	3.1%	6.8%	20.0%	+3.7	+13.2
AIME 2025†	base	30	1.2%	3.5%	13.3%	+2.3	+9.8
	TPT (rd 0)	30	4.0%	11.9%	20.0%	+7.9	+8.2

Table 7: Frontier API reference band (zero-shot plus the same verifier). Reported as a reference band, not a like-for-like ceiling (see Measurement integrity). The non-monotonicity across model sizes is the endpoints' own behavior.

model	GSM8K	MATH-500	AIME 2024	AIME 2025
gpt-5.5	99.0%	81.0%	100.0%	100.0%
gpt-5.4-mini	96.5%	74.0%	40.0%	43.3%
claude-haiku-4.5	98.0%	71.0%	26.7%	26.7%
qwen3.5-27b	94.5%	63.0%	93.3%	83.3%

N2 gains an OOD caveat. Scoring all six training variants on the two adjacent benchmarks (Table 8) refines N2: on GSM8K, cross-family and same-family distillation are close (the N2 null), but on MATH-500 they *separate*. Same-family `qwen-distill` reaches 73.6%/86.6% vs. cross-family `gpt-distill` 69.8%/81.0% (+3.8 Pass@1, +5.6 maj@16) and is the single most robust fine-tune. So N2's finding that teacher family barely matters holds on GSM8K but not out of distribution, where a same family teacher pulls ahead (n=100, so suggestive, not definitive). The verification step also keeps its value OOD: `tpt-nonprune` is the weakest variant on both benchmarks.

6 What Did and Did Not Reproduce

Table 9 collects the main results. The training axis lands near 88-92%; parallel test-time compute alone reaches a 27B's accuracy on a 9B; and a 2% confidence-routed escalation reaches 95.5% (within the single-seed noise band) with 98% of inference on-device. Restating the verdicts of Table 1: C1 and C2 *confirmed*, C4 *confirmed in kind*, and C3 *partially* confirmed (n=30) at 4-bit on 16 GB; N1 holds and generalizes to a coverage-dependent regularity; N2 holds (self-training ties frontier-teacher distillation up to a 27B teacher, a 397B teacher edges higher within noise, and the same-family edge appears out od distribution).

Three reproducibility takeaways follow. **(i)** A standardized harness is required, and any bespoke extractor must be audited against it (§5). **(ii)** Report maj@k *and* pass@k, since the gap between the realizable and oracle numbers is where routers and verifiers operate. **(iii)** At single-seed n=200, few-point differences are noise, and we say so wherever it matters.

7 Threats to Validity and Limitations

Single seed, wide intervals. All headline rows use one seed (3407); at n=200 the confidence interval is $\approx \pm 5\%$, so the TPT-vs-distill gap (88.5 vs. 89.5) and most few-point differences are not significant. The one robustness check we ran is an independent re-run of the maj@16 headline on the same GPU, which reproduced it within 0.2 points (93.8% vs. 94.0%).

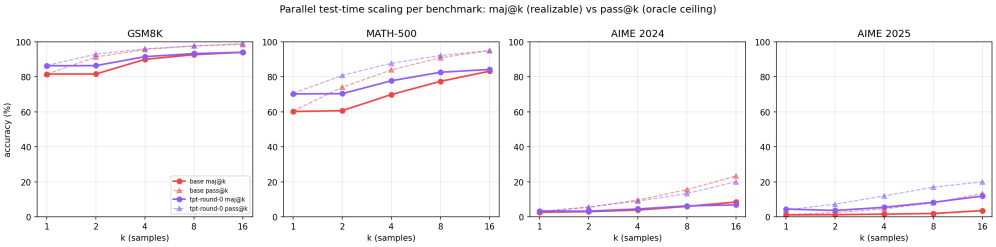

Figure 3: maj@k and pass@k curves across the difficulty gradient. Self-consistency keeps paying wherever oracle coverage stays high (GSM8K, MATH-500) and the base/TPT maj@k curves converge there (N1 holds OOD); on AIME, low coverage caps the realizable curve regardless of test-time compute.

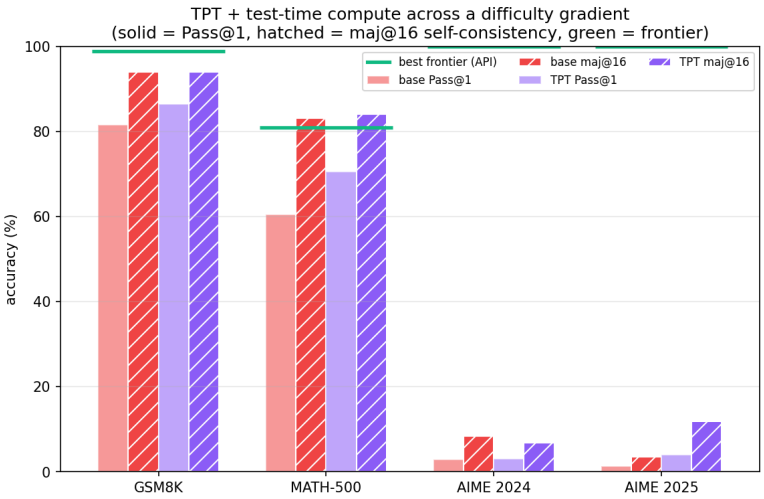

Figure 4: The difficulty gradient: local 9B (base / TPT, with maj@16) against the frontier reference band across GSM8K, MATH-500, and AIME. Near-parity on easy tasks; the gap reopens on the Olympiad tier.

Budget forcing is a base-only n=30 pilot; the per-run summary is committed (§4.3, App. B), so we report the numbers as a pilot. The s1-fidelity caveat (below) still applies.

GSM8K is already strong for the base; the generalization study (§5) partially addresses this, but MATH-500 (n=100) and AIME (n=30) are themselves small samples.

4-bit only. All local models are 4-bit; 16-bit might shift absolute numbers.

Protocol mismatch. The API tier is zero-shot+verifier vs. lm-eval 5-shot for fine-tunes; we treat it as a reference band, never a like-for-like ceiling. Separately, our headline TPT 88.5% is lm-eval 5-shot greedy and is *not* like-for-like with the original paper's temp-0.7 *sampled* Pass@1 (§3.1, Table 2).

Subset-resampled maj@k. maj@k is estimated by resampling size-k subsets from K=16 draws, not independent draws at each k.

s1 extend not faithful. Our "Wait" padding imposes a *minimum* think budget that s1 lacks and mixes up s1's two separate knobs (§4.4, Table 2); the curtail path *is* faithful. The minimum-budget forcing over-thinks easy problems, so an adaptive budget would likely do better than a fixed-B schedule.

Latency/cost. Escalation latency via API is network-dependent; we report compute counts and a throughput-based estimate, not production SLAs.

Table 8: Deferred comparison: all six training variants on the two adjacent benchmarks, same maj@k protocol. Same-family `qwen-distill` is the most robust fine-tune OOD; the same-family advantage shows up out of distribution (refining N2).

model	GSM8K Pass@1	GSM8K maj@16	MATH-500 Pass@1	MATH-500 maj@16
base	81.5%	94.0%	60.4%	83.2%
tpt-round-0	86.5%	94.0%	70.6%	84.3%
tpt-round-1	88.3%	94.5%	70.4%	82.1%
tpt-noprune	84.4%	93.5%	66.4%	80.5%
gpt-distill	**89.7%**	94.6%	69.8%	81.0%
qwen-distill (27B)	86.7%	**96.7%**	**73.6%**	**86.6%**

Table 9: Head-to-head on GSM8K test. Training plateaus near 90%; parallel TTS reaches 94%; a thin confidence-routed escalation *reaches* 95.5% (within the single-seed noise band) while keeping 98% of work on a 16 GB card. The base row is a T=0.8 *sampled* maj@1, whereas the two greedy training rows are lm-eval 5-shot. The two use different harnesses and should not be cross-compared. Italic rows are reference models that do not fit the card.

approach	axis	GSM8K	inference compute	on 16 GB?
base, maj@1 (T=0.8 sampled)	n/a	81.4%	1×	yes
TPT round 0, lm-eval 5-shot greedy	training	88.5%	1×	yes
same-family 397B distill, lm-eval 5-shot greedy	training	92.5%	1×	yes
base, maj@16	parallel TTS	94.0%	16×	yes
TPT, maj@16	parallel TTS	94.0%	16×	yes
cascade: TPT maj@16 + 2%→27B	TTS + routing	**95.5%**	16× local, 2% big	**98% local**
cascade: TPT maj@16 + 6%→27B	TTS + routing	96.5%	16× local, 6% big	94% local
cascade: TPT maj@16 + 24%→GPT-5.5	TTS + routing	98.5%	16× local, 24% big	76% local
Qwen3.5-27B (reference)	bigger model	*94.5%*	big	no
GPT-5.5 (reference)	frontier	*99.0%*	frontier	no

8 Conclusion

On a single 16 GB consumer GPU, a 4-bit 9B can reach the GSM8K frontier *band*, but not by training alone. The standard small-model reliability methods (verified self-training, parallel and sequential test-time scaling, confidence routing) all replicate at 4-bit on 16 GB, confirming C1–C4. The durable lessons, however, are not the 95.5%. They are the substitution effect (N1), where fine-tuning and test-time compute buy overlapping reliability, now sharpened into a coverage-dependent regularity; and the distillation comparison (N2), where teacher strength and family barely move the needle once the data is verified and on-task. Even a much larger 397B teacher only edges a few points higher (92.5%), and the same-family teacher's one clear win shows up out of distribution, where its distillation is the more robust fine-tune. The practical recipe is to *compose*: a self-trained, format-clean 9B; maj@k parallel sampling; vote-agreement routing; and a thin escalation tier. Promising next steps from the reference set are a temperature *ladder* for the sampling tier (Wu et al., 2025) and a *trained* verifier (CoVer authors – confirm before camera-ready, 2026) (a robotics/VLA result we apply by analogy to math reasoning) to capture the +5-point maj@k→pass@k headroom.

Reproducibility Statement

Every claim binds to an exact script in the released harness. Training axis: `run_tpt.sh` (TPT loop: `tpt_sample/tpt_prune/tpt_make_json/tpt_train/tpt_eval`), `run_gpt_distill.sh`, `run_distill_qwen.sh` (teacher set by `QWEN_DISTILL_MODEL`; the 397B run uses `QWEN_DISTILL_DIR/_MERGED/_ADAPTER` for isolated paths), `eval_lm_harness.py`, and the API ceiling `api_eval.py`. Inference axis: `run_tts.sh` (parallel + sequential), `cascade_big_eval.py` + `cascade_analyze.py` (routing), `make_tts_charts.py` (figures). Generalization: `run_multibench.sh`,

`bench_sample.py`, `bench_budget.py` (its per-run `*_budget_summary.json` are committed), `bench_api_eval.py`, `bench_analyze.py`. Shared verifier: `gsm8k_verify.py` (and `benchmarks.py` for the OOD benchmarks); the voting arithmetic is unit-tested in `test_tts_vote.py`. Fixed seed 3407; environment is CUDA 12.8 with Unsloth 4-bit; model revision, few-shot templates, stop sequences, and data splits are pinned in the repository `README`. *Open item:* all results are single-seed (3407), and the $\approx \pm 5\%$ intervals are binomial estimates at n=200, not measured across repeated runs. We did not run multiple seeds; the only robustness corroboration is an independent re-run of the maj@16 headline (93.8% vs. 94.0%). Multiple seeds with measured CIs remain the main upgrade for a future revision.

References

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Caia Costello, Simon Guo, Anna Goldie, and Azalia Mirhoseini. Think, prune, train, improve: Scaling reasoning without scaling models. *arXiv preprint arXiv:2504.18116*, 2025.

CoVer authors – confirm before camera-ready. Scaling verification > scaling policy: Test-time verification for reasoning (CoVer). *arXiv preprint arXiv:2602.12281*, 2026.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized LLMs. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. arXiv:2305.14314.

Leo Gao, Jonathan Tow, Baber Abbasi, et al. A framework for few-shot language model evaluation. https://github.com/EleutherAI/lm-evaluation-harness, 2024. Zenodo, version 0.4.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks*, 2021. arXiv:2103.03874.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022. arXiv:2106.09685.

Hugging Face. Math-verify: A robust mathematical expression evaluation system. https://github.com/huggingface/Math-Verify, 2025.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*, 2025.

Jon Saad-Falcon et al. OpenJarvis: On-device language agents matching cloud accuracy via decomposition and routing. *arXiv preprint arXiv:2605.17172*, 2026. Author list to confirm before camera-ready.

Scaling Intelligence Lab. Think, prune, train (official code). https://github.com/ScalingIntelligence/TPT, 2025.

simplescaling. s1: Simple test-time scaling (official code). https://github.com/simplescaling/s1, 2025.

Unsloth AI. Unsloth: Fast and memory-efficient LLM finetuning. https://github.com/unslothai/unsloth, 2024.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations (ICLR)*, 2023. arXiv:2203.11171.

Wu, Azalia Mirhoseini, and Thierry Tambe. On the role of temperature sampling in test-time scaling. *arXiv preprint arXiv:2510.02611*, 2025. Author list to confirm before camera-ready.

A Full per-k maj@k / pass@k tables

GSM8K per-k curves are summarized in Table 4; full per-k values for all benchmarks (including maj@1, maj@2) are produced by `bench_analyze.py` into `outputs/multibench/multibench_self_consistency.md`. The complete per-k grids (maj@1 through maj@16 and pass@1 through pass@16, all benchmarks) are in that committed file.

B Budget forcing: pilot results

The s1 budget-forcing pilots are small (n=30, single seed) and our extend mechanism deviates from s1 (§4.3, Table 2), so we report them as pilots, not headline numbers. The per-run summaries (`*_budget_summary.json`) are committed to the released harness, so the numbers are independently re-derivable. Accuracy rises monotonically with the think-token budget on every run (Table **??**); on MATH-500 the self-trained model overtakes the base at the largest budget (90.0% vs 86.7%)

C Prompt templates and stop sequences

Local inference uses the few-shot `Question:/Answer:` recipe with a stop sequence; the self-training generation prompt is "Solve this step by step. End with: Final answer: `<number>`". Exact few-shot exemplars and stop strings are pinned in the repository. The verbatim few-shot exemplars and stop strings are in benchmarks.py and the repository README.