Chasing the Small-Frontier on a Fixed Rig

Metric Artifacts, Benchmark Contamination, and a Precisely Measured Gap

Jovonni L. Pharr
Georgia Cyber Warfare Range
info@gacwr.org

September 2026

Every number in this paper is emitted by a committed analysis script from committed data, and is reproducible on a machine with no GPU at all.

Abstract

This work is a measurement-driven account of training a small language model toward the quality of the small-frontier on a fixed pair of graphics processors, with no recourse to rented compute. Its contribution is a method and the results that method produced: adversarial self-audit of a laboratory's own instrument, which here recovered more capability-relevant signal than additional training would have. We establish, first, that a composite averaging unnormalized per-task margins is dominated by a single high-variance task, and that one benchmark in particular is a class-prior coin flip whose apparent victory is an artifact of that prior. Scoring the reference through our *own* harness then exposed a deeper fault—the scorer computed a different quantity for the two models and normalized by token count across incomparable vocabularies—so we rebuilt it around a single byte-normalized, joint-tokenization scorer, which is what makes a cross-vocabulary comparison legitimate at all. On that instrument the picture is precise and split: the reference leads argmax accuracy by 0.028 (95% CI $[+0.002, +0.055]$), yet on the continuous log-likelihood margin our larger model leads. The lead is thinner than it looks—only one of seven per-task differences is individually significant—and it is measured against a reference that saw roughly a thousand times more training tokens than we did, so it is a statement about scale at least as much as about skill. We measure, second, what the optimizer's peak learning rate is actually worth on this rig. Re-derived from the sweep's own per-example margins, an apparent threefold advantage is carried entirely by the single benchmark Section 3 identifies as a coin flip. Excluding it the factor is 1.16, its confidence interval includes zero, and the training objective—bits-per-byte—orders the four arms the other way, with the long-standing default best of all: on the evidence available, the learning rate is not the lever at this scale, and we specify the seeded sweep that would settle its size. We establish, third, that the corpus this project was preparing to *manufacture* at a cost of years of compute is freely available to *download* at the cost of a day. We document the method of adversarial multi-agent auditing that produced these results, and the plan they support.

Part III adds the first benchmark-contamination measurement of its kind in this project: on our own corpora *and*, to our knowledge for the first time in the small-model literature, on four components of a widely used reference model's published training mixture, with a detector validated against positive and negative controls built from the corpora themselves (64/64 and 0/64). The result is counter-intuitive and it is the paper's sharpest: contamination is concentrated *not* in web data but in the small corpus we hand-assembled for quality and inject at the annealed end of every run. At a matched scanning budget it contains verbatim thirteen-grams of 17 of 4000 evaluation items where every web corpus returns zero or one. Scanned to completion, one component of the reference's own published mixture reaches 5.0% on BOOLQ and 2.8% on MMLU — a level at which a decontamination statement belongs in every small-model report. We then localise the learning-rate effect to the level of individual examples: seven of the eight benchmarks' margin distributions are unchanged between the two arms and one translates bodily, and the obvious alternative mechanism —

that the deciding benchmark's two-byte answers put it on a wider scale — is measured and refuted. Finally we mine the project's seventy-six-run record: $L(N, D)$ fitted on it misses its own pre-registered held-out bar, so we publish the forecast error rather than a forecast, and the same ledger yields a receipt-backed data point the emergence debate rarely gets — over a fifteenfold parameter range on one rig, no per-task capability threshold appears at all. Tasks are cleared from the smallest model in the record or never cleared.

Contents

Part I
Preliminaries

1 The State of Today

There exists a class of small language models—on the order of one hundred to five hundred million parameters—that attain a surprising fraction of the competence of their larger relatives

on standard benchmarks. The existence of these models is a standing proof that a great deal of capability is reachable at small scale, given the right data and the right recipe. It is natural, for a laboratory constrained to modest hardware, to take one such model as a reference and to ask what separates one's own model of comparable size from it. The reference we adopt is a widely reported one-hundred-and-thirty-five-million-parameter model; our own model is slightly larger, at one hundred and fifty-eight million parameters, and yet it scores lower on our own evaluation harness.

It is tempting to read that as evidence that the difference is one of skill and recipe rather than of scale. The reading does not survive its own denominator. Our champion's lineage has seen 1.97×10^9 training tokens—12.4 tokens per parameter, well below the compute-optimal ratio—while the reference was trained on the order of 2×10^{12}, a factor of 1017 more (Table 1). A smaller model trained on a thousand times more data outperforming ours is, in the first instance, a *scale* statement. The measurement that separates the two accounts is an iso-token baseline—a parameter-exact public model at a matched token budget—and it is specified in §12. The premise that organizes this investigation is therefore the one the measurements settle: our own measuring instrument was wrong in ways we could find without spending a single additional GPU-hour, and finding them changed the answer.

Table 1: Training budget. Our figures are recovered by walking each run's warm-start pointer back to the from-scratch root. The reference's budget is as reported by its authors.

	params	tokens seen	tok/param	frac. of Chinchilla	wall-clock (this rig)
ours (v412 lineage)	158.6M	1.97×10^9	12.4	0.62	26.5 h
reference (SmolLM2-135M)	135M	$\sim 2 \times 10^{12}$	$\sim 1.5 \times 10^4$	$\sim$740	—
ratio (reference / ours)	0.85×	$\sim$**1017**×	$\sim$1200×		

2 The Rig

The constraint is absolute and it is worth stating precisely, for it shapes every decision that follows. The laboratory possesses two graphics processors and no others; there is no cloud, and there will be no additional hardware. Local computation is, in the sense of marginal cost, unlimited—there are no rental bills—but it is not unlimited in *throughput*: a single processor sustains on the order of seventy-four million training tokens per hour, and the trainer is not data-parallel. Models are capped at one gigabyte. Within these constraints the question is not how to buy one's way to the frontier but how to reach it by skill.

Part II
Diagnosing the Instrument

3 The Ruler, and What It Was Actually Measuring

The first and most consequential finding concerns not the model but the ruler. Our headline metric is a composite that averages, across a set of tasks $\mathcal{T}$, a smooth per-example margin between the log-likelihood assigned to the correct completion $y^\star$ and the mean log-likelihood

assigned to its distractors $y \in D(x)$:

$$\text{Composite} \;=\; \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} \underbrace{\mathbb{E}_{x\sim t}\left[\log p_\theta(y^\star \mid x) - \frac{1}{|D(x)|}\sum_{y\in D(x)} \log p_\theta(y \mid x)\right]}_{\text{margin}_t}. \tag{1}$$

This composite has a structural defect: the per-task margins margin_t are not normalized to a common scale, and a task whose margins are intrinsically large dominates the unweighted average. Figure 1 decomposes the composite of our champion by task and shows the pathology directly—the margin of a single science task, SCIQ, is 0.230 out of a total of 0.988: 23% of the aggregate from one of thirteen tasks, and 3.0× the mean task's contribution.

The domination is real and it is bounded: the other twelve tasks sum to 0.758, and ARC_EASY (0.154) and HELLASWAG (0.138) are of the same order as SCIQ. A single task carrying 23% of an unweighted thirteen-task average, at 3.3× the next largest contribution, is a scale-domination defect sufficient to motivate the fix.

One further distinction, material to everything that follows: there are *two* composites in this project. The one decomposed in Figure 1 spans thirteen tasks including SCIQ, and is the one SCIQ dominates. The one used for every learning-rate number in Section 4 spans eight tasks and does *not* contain SCIQ at all; its dominant task is BOOLQ. Diagnosing a defect in the first aggregate and then reporting headline results on the second is how a defect of this kind escapes notice, and it is why every number in Section 4 states which composite it belongs to.

A more serious defect afflicts one particular benchmark, BOOLQ, a yes-or-no reading task scored by the likelihood of the single tokens "yes" and "no." Every model answers largely according to its *class prior*. Writing a_+ and a_- for accuracy on the true and false instances, the reported accuracy on a majority-+ set of positive fraction π is $\pi a_+ + (1-\pi)a_-$, which a prior-only classifier maximizes by always emitting the majority label. The measure that is invariant to π is the balanced accuracy

$$\text{BA} \;=\; \tfrac{1}{2}\big(a_+ + a_-\big), \tag{2}$$

which we *expect* to sit at essentially $\frac{1}{2}$. Eq. (2) is stated as a prediction, with the experiment that discriminates it fixed in advance (§12): both scorers now record per-class and balanced accuracy on every task, balanced accuracy inside $[0.48, 0.52]$ substantiates the prediction, and anything above 0.55 refutes it.

What *is* measured is the instability. Across the 76 logged runs of the pinned snapshot, reported BOOLQ accuracy has standard deviation 0.089, while a genuine four-way task, ARC_EASY, has 0.047 (Figure 2); BOOLQ is 1.9× as variable, not three times.

Two properties of that distribution matter and they are easy to conflate. Excess variance is one; displacement is the other, and it is the sharper signature. BOOLQ accuracy does not swing about the 0.50 chance line: its median across runs is 0.614 and 79% of runs sit *above* chance. A distribution displaced above chance is precisely what a majority-class prior produces. Because the figure's inputs are frozen to a committed snapshot, both numbers are stable under re-derivation — an aggregate computed over a growing leaderboard is not, and this one was pinned for that reason. The 76 runs span 10.3 to 158.6M parameters and three architectures, so their spread mixes genuine capability variance with metric noise; the per-class measurement specified above is what isolates the prior.

The apparent victory of our champion on BOOLQ is, on this reading, an artifact of a prior; it simultaneously inflated our champion's standing and concealed the true shape of the gap to the reference.

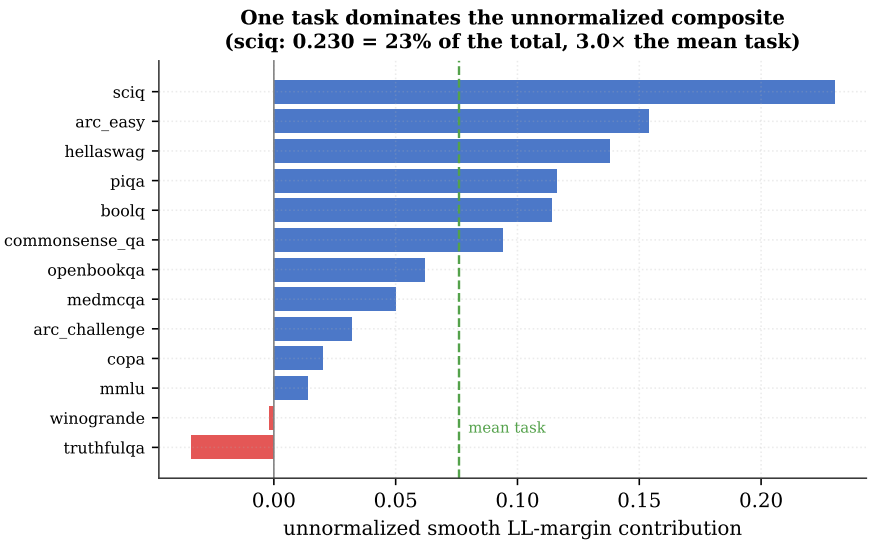

Figure 1: Task decomposition of the champion's thirteen-task composite (Eq. 1). SCIQ contributes 0.230 of a 0.988 total—23% from one task, 3.0× the mean task (dashed line). The aggregate is scale-dominated, though not by as much as all other tasks combined. Reproduced from the per-task margins of the pinned snapshot.

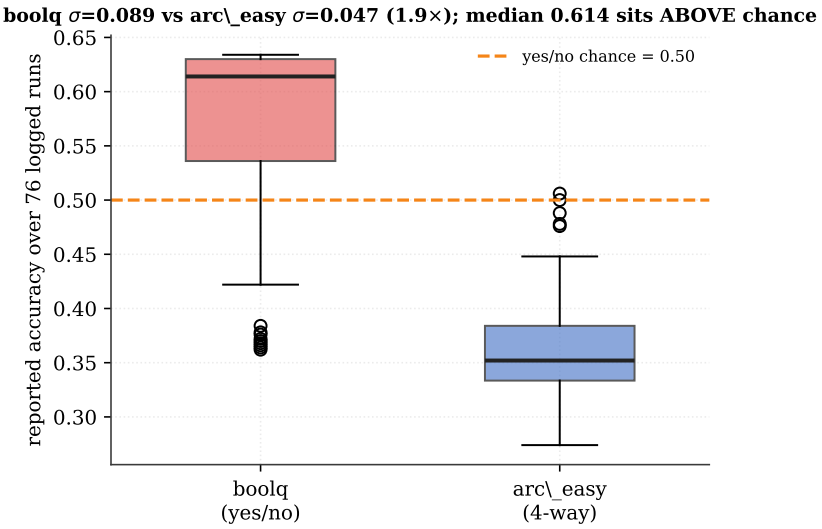

Figure 2: BOOLQ accuracy over the 76 pinned runs is 1.9× as variable as ARC_EASY ($\sigma = 0.089$ vs 0.047) and sits *above* the yes/no chance line (median 0.614; 79% of runs above 0.50)—the signature of a majority-class prior. Note that these runs span a 15× parameter range and three architectures, so this spread is suggestive, not dispositive; the dispositive per-class test is queued (§12).

Removing the coin-flip benchmark and giving the remaining tasks equal weight is necessary but not sufficient, for it corrects only our own scores. Scoring the reference through our own harness is the next step—but it exposed a deeper fault, and it is the fourth correction: *the ruler itself was miscalibrated.* Our two scoring paths did not compute the same quantity. Our own checkpoints were scored by encoding context and answer separately and concatenating the token identifiers, which retokenizes the answer as though it began a document; the reference was scored by a joint encoding whose boundary index silently dropped the first answer token

on multi-token answers; and both normalized by *token count*, a quantity that is not comparable across a sixteen-thousand and a forty-nine-thousand vocabulary. We replaced both with a single shared scoring *function*: one joint tokenization, an answer boundary recovered by longest common prefix, and normalization by the answer's *byte* length, which is tokenizer-invariant.

Two qualifications on the word "shared." The scoring function is shared; the numerical precision is not—our own checkpoints are evaluated in `float32` (the checkpoint loader applies no cast and no autocast) while the reference was evaluated in `bfloat16`, whose eight mantissa bits are of the same order as several of the per-task margins we report (e.g. WINOGRANDE, +0.010). And the two columns of Table 2 were measured at different sample sizes, $n = 300$ for ours and $n = 500$ for the reference, so the comparison is *unpaired*. Both are addressed by the matched-n, matched-precision re-run reported in item 1 of §12; the driver now takes an explicit precision flag and writes per-example margins, so the comparison can be repeated at any n on either device. Subject to those two qualifications, this is the ruler on which the comparison below is made.

The result on that ruler is precise and it is split. On argmax accuracy the reference leads, but the two metrics *disagree* (Table 2, Figures 3 and 4): over the seven tasks that survive the BOOLQ deletion the reference leads mean accuracy 0.405 to 0.377, a gap of 0.028 with 95% CI $[+0.002, +0.055]$ ($p = 0.035$); yet on the byte-normalized smooth log-likelihood margin over those *same* seven tasks, *our* model leads, +0.188 to +0.141.

Three properties govern how much that comparison can bear, and each is a rule applied to every aggregate here. *(i) Every cell traces to scorer output.* Both rows of Table 2 are read from logged evaluation runs at the n each was run at, and the plotting scripts read those measurements rather than any literal; a benchmark row that exists only inside a plotting script is not a measurement, and the difference between the two is up to 0.033 per task — larger than the headline gap of 0.028 it would feed. *(ii) Both metrics span one task list.* Accuracy and margin are reported over the same seven tasks throughout. The distinction is not cosmetic: BOOLQ supplies our single largest positive margin (+0.261), so an eight-task margin next to a seven-task accuracy would read as +0.197 vs +0.163 and flatter us by exactly the benchmark this paper has just set aside. On the seven-task list our margin lead is +0.188 to +0.141, and our model does lead. That margin remains unnormalized and is now dominated by COMMONSENSE_QA rather than by SCIQ, so it inherits the defect Eq. (1) diagnoses; the chance-corrected form prescribed in Section 3 is the natural next form of it. *(iii) A count of task wins is not a result without its sign test.* The reference leads *five* of seven (sign test $p = 0.45$), and with 95% confidence intervals on each difference exactly *one* — ARC_EASY, +0.087 $[+0.015, +0.158]$ — excludes zero. The other six differences are not distinguishable from noise, and on three tasks (MMLU, ARC_CHALLENGE, WINOGRANDE) at least one of the two models is statistically indistinguishable from chance. Reporting the tally without the interval would have converted a coin-flip margin into a sweep.

A model that trails on the discrete decision while leading on the underlying margin invites a tempting reading: that its knowledge is close and only its calibration at the decision boundary lags. That reading is a hypothesis about the decision rule, and item 2 of §12 tests it directly, by fitting the length-normalisation exponent and the per-token and per-byte priors *before* the argmax, per task, cross-validated — and by fitting the identical family to the reference, since a claim that *our* model is merely miscalibrated requires that recalibration help us more than it helps them.

The scope of the multiple-choice comparison is worth stating exactly, because multiple-choice log-likelihood ranking is the one family of metrics on which our model is close and it is not the only family in the file. Read against the pinned snapshot: v412 scores 0.0 on HumanEval PASS@1, 0.0 on MBPP and 0.0 on GSM8K; HumanEval PASS@1 is 0.0 for *every* one of the 76 runs; GSM8K is non-zero for 34 of the 76 and MBPP for 4 of the 76, at values of {0.0167, 0.0333, 0.05, 0.0833} on GSM8K — one to five items correct — and at most 0.02 on MBPP. The exact statement is therefore: **no model this project has trained exceeds** 0.083

on grade-school mathematics or 0.02 **on MBPP, and none has ever solved a single HumanEval problem**. Generation is a different regime from ranking at this scale, and §12 states the matched generative table that would put both models in it on the same footing.

Table 2: Head-to-head, both models scored by the same byte-normalized function. The reference row is read from a logged evaluation run at $n = 500$ in `bfloat16`; our row is scored by the same function at $n = 300$ in `float32`. The two n's and the two precisions differ, so this comparison is *unpaired* and precision-mismatched; item 1 of §12 repeats it with both matched. Brackets are 95% CIs on the difference (unpaired two-proportion); $*$ marks the one difference that excludes zero. Accuracy and margin use the *same* seven-task list.

	MMLU	ARC_E	ARC_C	HELLA	WINO	OBQA	CSQA	**mean**
ours (v412, 158M), n=300	0.290	0.483	**0.283**	0.367	0.517	**0.333**	0.367	0.377
reference (135M), n=500	**0.340**	**0.570***	0.272	**0.422**	**0.518**	0.328	**0.388**	**0.405**
gap (ref − ours)	+.050	+.087	−.011	+.055	+.001	−.005	+.021	+.028
95% CI (lo)	−.016	+.015	−.075	−.014	−.070	−.073	−.048	+.002
95% CI (hi)	+.116	+.158	+.053	+.125	+.073	+.062	+.091	+.055
chance level	.25	.25	.25	.25	.50	.25	.20	—
smooth margin, ours	+.083	+.353	+.086	+.130	+.010	+.146	+.509	+.**188**
smooth margin, reference	+.062	+.251	+.041	+.096	+.010	+.117	+.408	+.141

The reference leads five of seven tasks (sign test $p = 0.45$) and exactly one of the seven per-task differences is individually significant. On the seven-task mean the gap is significant ($p = 0.035$), but its lower confidence bound, +0.002, is smaller than the harness-version sensitivity quantified below—so the exact summary is that the reference is ahead, by an amount this measurement can just resolve.

How much of 0.028 **is the ruler?** This is the question a cross-model benchmark number should always be asked and almost never is, so we measure it. Scoring a *fixed* checkpoint under different generations of our own harness moves it by more than the gap being reported. The same SmolLM2-135M weights score mean accuracy 0.384 under the pre-fix scorer and 0.434 under the post-fix one—a swing of 0.050; a second fixed checkpoint moves by up to 0.067 on a single task and its composite by a factor of 2.2; the same v412 checkpoint moves by up to 0.118 on a single task and its composite by a factor of 2.1. Every one of these exceeds 0.028. This does not invalidate the comparison, because both columns of Table 2 use the same post-fix scorer—but it does establish that any such number is a property of a model *and* a ruler, which is why we report the confidence interval rather than the point estimate, and why a benchmark gap of this size published without its instrument's variance is not interpretable.

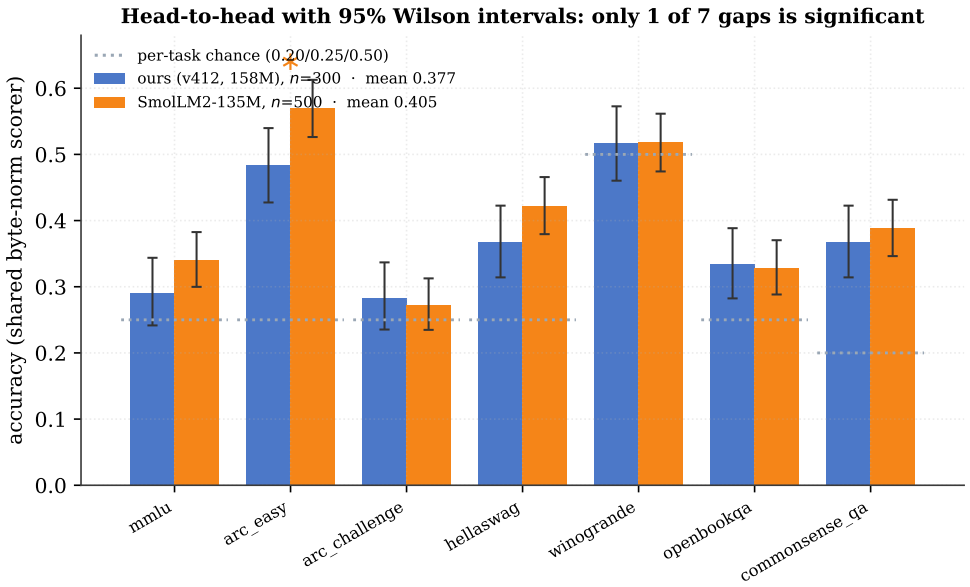

Figure 3: Head-to-head on our fixed byte-normalized harness, with 95% Wilson intervals and *per-task* chance lines. Per-task chance lines matter: a single line drawn at 0.25 across all seven makes the WINOGRANDE column—where both models are at chance—read as the strongest on the chart. The reference leads five of seven, and only ARC_EASY (∗) is significant. Both columns are generated from one logged measurement each; neither is a literal.

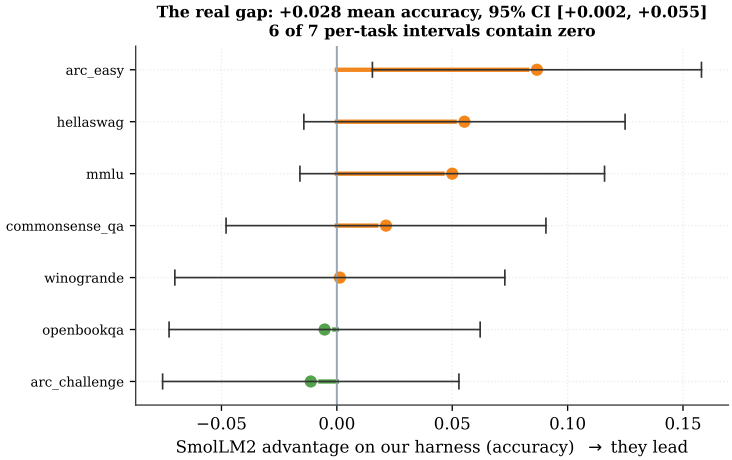

Figure 4: The real per-task gap, with 95% confidence intervals on each difference. Positive means the reference leads. Six of the seven intervals contain zero: the advantage is diffuse in the sense that no single column carries it, but it is also, per task, mostly undetectable at this sample size.

4 What the Peak Learning Rate Is Actually Worth

The optimizer's peak learning rate is the cheapest knob a laboratory owns, and on the aggregate this project used to grade with, it looks like the most valuable. A controlled sweep of η describes a clean inverted-U in the smooth metric $M(\eta)$: the value every one of our runs had used—including the historical champion—sits at $\eta = 6{\times}10^{-4}$, on the cold side of an apparent optimum

$\eta^\star \approx 1.5\times10^{-3}$, so that

$$\frac{M(\eta^\star)}{M(6\times10^{-4})} \;=\; \frac{0.204}{0.069} \;\approx\; 3.0 \qquad \textit{(eight-task aggregate; see Eq. (4)).} \tag{3}$$

A threefold return on one hyperparameter would be the most consequential result on this rig, and it would license a sweeping consequence: that every previous verdict of the form *this data intervention does not help* was rendered at the wrong learning rate. A claim of that reach earns the most sceptical test we can build, so this section applies one to it.

Where the factor of three lives. The metric M in Eq. (3) is the eight-task composite, and it includes BOOLQ—the benchmark Section 3 of this same paper declares a class-prior coin flip. Recomputing from the sweep's own per-example margins ($n = 1000$ per task, saved on disk for all four arms) and simply deleting that one task:

$$\begin{aligned} \frac{M_{\setminus\text{BOOLQ}}(1.5\times10^{-3})}{M_{\setminus\text{BOOLQ}}(6\times10^{-4})} \;&=\; \frac{0.1408}{0.1214} \;=\; 1.16, \\ \Delta = +0.0195, \qquad \text{CI}_{95}\,[-0.0014,\, +0.0410], \qquad & p = 0.067, \end{aligned} \tag{4}$$

by paired bootstrap over the same examples ($B = 5000$). The interval contains zero. The other two arms are flatly null: 2.5×10^{-3} gives $\Delta = +0.0011$ ($p = 0.90$) and 4×10^{-3} gives $\Delta = -0.0083$ ($p = 0.46$). The entire factor of three lives in BOOLQ, whose accuracy across the four arms is 0.389/**0.624**/0.392/0.374: a lone yes-bias outlier at exactly the arm we crowned.

Three further facts point the same way.

1. **The training objective disagrees outright.** Held-out bits-per-byte for the four arms is 1.3835/1.3936/1.3972/1.4033. The prior default, 6×10^{-4}, has the *best* loss of all four. An inverted-U in a downstream proxy that is monotone the other way in the quantity actually being optimized is a warning, not a discovery.
2. **Argmax accuracy also disagrees.** The seven-task ex-BOOLQ accuracies are 0.307, 0.305, 0.303, 0.303: flat, with the prior default nominally highest.
3. **The sweep is not budget-matched, as a claim of this reach requires.** The arms were matched on *wall clock* (45.0 min each), not on tokens, so the 6×10^{-4} arm completed 6527 steps against the claimed optimum's 6409: 1.9% *more* tokens for the arm we called inferior. And there is one run per learning rate: the trainer had no seed control at all, so no run in this project has ever been repeated, and every Δ above is being compared against a run-to-run standard deviation that does not exist.

What the sweep supports. At matched wall clock and a single seed, a peak learning rate of 1.5×10^{-3} is *directionally* better than 6×10^{-4} on the ex-BOOLQ smooth margin by 1.16×, and that difference is not statistically significant. It is contradicted by both bits-per-byte and argmax accuracy. On the evidence this sweep can carry, the learning rate is not a 3× lever on this rig, and no back catalogue of data-intervention verdicts needs re-running: a consequence that expensive requires an effect that survives its own arithmetic, and this one does not. The generalisable form of the result is the one worth carrying away — a downstream aggregate can manufacture a large effect out of a single class-prior task, and the training objective is the check that catches it.

What remains open is the size of the real effect, and one external datum bears on it: the reference model's technical report discloses a peak learning rate of 3×10^{-3} for its 135M model, five times hotter than our historical default. That is genuine evidence that our default may be cold; it is not evidence about the size of the effect on our rig. The trainer now takes an explicit seed and a token-matched step budget, so the properly powered sweep — five learning

rates, three seeds each, token-matched, graded on ex-BOOLQ margin *and* bits-per-byte, against a pre-registered 2σ bar — is specified and ready to run (§12). It will also produce the first run-to-run noise floor this project has ever had.

Table 3: The peak-learning-rate sweep, read on four instruments at once. The grey row is the eight-task aggregate including BOOLQ; the row below deletes that one task and the effect largely vanishes. Bits-per-byte—the quantity actually being optimized—is best at the long-standing default, and so is seven-task argmax accuracy. Wall-clock matched at 45.0 min, one seed per arm.

Muon peak η	6×10^{-4} *long-standing default*	1.5×10^{-3} *apparent opt.*	2.5×10^{-3}	4×10^{-3}
8-task comp., *incl.* BOOLQ	0.069	**0.204**	0.075	0.026
7-task comp., *ex* BOOLQ	0.121	**0.141**	0.123	0.113
vs. default (p)	—	+.020 (.067)	+.001 (.90)	−.008 (.46)
7-task argmax accuracy	**0.307**	0.305	0.303	0.303
held-out bits-per-byte ↓	**1.3835**	1.3936	1.3972	1.4033
BOOLQ accuracy	0.389	**0.624**	0.392	0.374
steps completed	6527	6409	6407	6404

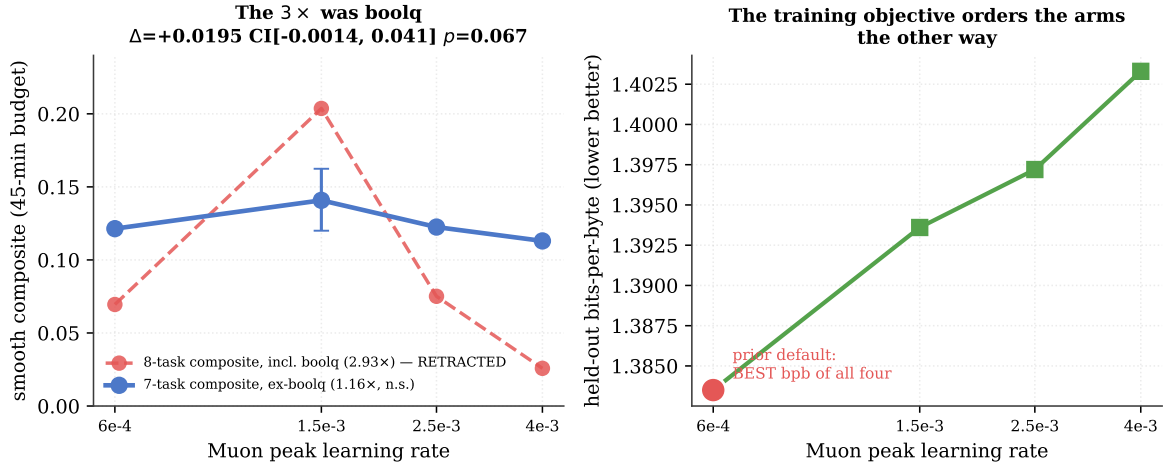

Figure 5: *Left:* the eight-task inverted-U (red, dashed) and the same sweep with BOOLQ deleted (blue, solid, with the paired-bootstrap interval on the peak). The factor of three is one task. *Right:* held-out bits-per-byte, the quantity the optimizer actually minimizes, which is *best* at the long-standing default.

5 One Should Download, Not Manufacture

The third result concerns the acquisition of data, and it is the one with the largest practical return. The measured deficit is at bottom a deficit of knowledge and reasoning, and the natural response is to supply that as data. This project was preparing to *generate* it, by distillation from a local teacher model. Two independent adversarial audits arrived at the same objection: the generation is infeasible and the manufactured data would be poor. A teacher of moderate size emits on the order of one million tokens per hour; the tens of billions of tokens required would consume years of compute, and the output of a weak generator is itself weak. The corpus we needed was, in fact, the reference model's own training data—curated synthetic textbooks, mathematics, and filtered web text—and it is openly published and re-tokenizable to our vocabulary at the cost of a day of ordinary input and output. The generalisable rule

is worth stating plainly, because it is the highest-return decision in this paper and it cost no compute to reach: at the small-model scale, check what has already been published before budgeting to manufacture it. The audit converted years of planned compute into a day of download.

6 What More of Our Own Data Bought

We ran a token-scaling series over our single narrow corpus and measured, for each leg, the held-out bits-per-byte and a fair nine-task equal-weight accuracy. Bits-per-byte falls from 1.1610 to 1.1435 while accuracy does not move, holding at 0.388 with a spread of 0.003 across the six legs (Figure 6). Compression improves; measured competence does not.

Four properties of this series determine what it shows, and three of the four change the answer. Each is recovered from the training logs rather than from the experiment's design document, and the difference between those two sources is the point.

It is one warm-start chain, not six independent models. Every leg warm-starts from the previous leg's final checkpoint, so this is one sequential chain, not a series of controlled arms. The peak learning rate declines monotonically *along the x-axis*—1×10^{-3}, 8×10^{-4}, 8×10^{-4}, 7×10^{-4}, 6×10^{-4}—so the paper's own second headline, whatever its true size, runs exactly parallel to the independent variable. Every leg also carries a logit-distillation objective ($\lambda = 0.5$) and injects a *second*, different corpus over the last 40% of its budget, so "the same corpus" is inaccurate as well. And the budgets are unequal: the final leg ran 180 minutes against the others' 240, so the ticks are not evenly spaced in tokens.

No data was ever repeated. This is the most consequential property of the design. The data-constrained scaling literature—"past a few epochs, the marginal capability of an additional pass over a fixed corpus decays to zero"—is the natural mechanism to reach for here, and it does not apply. The corpus file holds 9.14×10^9 tokens; the whole series consumed 1.58×10^9 of them. That is **0.17 epochs**. Not one token was seen twice. The repeated-data regime was never entered, so that literature cannot explain anything observed here, and §13 places it accordingly. What the series actually varies is the number of *fresh* tokens drawn from one narrow distribution, which is a distribution-diversity observation, not an epoch one.

The terminal point has two disagreeing sources. The project leaderboard records 1.126 for the final leg. That value does not appear anywhere in that run's training log, whose last recorded evaluation is 1.1435. The other five legs agree between log and leaderboard to $\sim$0.001; this one differs by 0.018, eighteen times as much — and it is the single point that decides whether the curve is monotone. A curve drawn from two sources is not a curve, so Figure 6 is plotted from the training logs alone, on which it *rises* at the last point: $1.1610 \to 1.1423 \to 1.1435$. Bits-per-byte falls, but not monotonically. Re-evaluating that checkpoint (§12) resolves the disagreement; until it does, the leaderboard value is drawn as a dotted line and not used.

"Flat" is a null statement, and it needs its power. With nine tasks at 300 items and accuracy near 0.39, the minimum effect this design can detect at 80% power and $\alpha = 0.05$ is **0.037**—larger than the entire 0.028 gap to the reference that the paper is trying to close. The observed spread, $\sigma = 0.0031$, is far below that floor, so the exact statement is not that accuracy is flat but that *no effect was detected at our power*. On the project's own gated smooth metric the same conclusion holds with intervals wide enough to contain a +0.05 gain. We therefore report an absence of detection with a stated detection limit rather than a demonstrated null. Every null in a small-model report deserves the same treatment, and most do not get it.

What the series supports. Read within those four bounds, the observation is this: adding 1.6 billion fresh tokens from one narrow web corpus, along a warm-start chain with a declining learning rate, improved held-out compression by about 1.5% and moved measured accuracy by less than we could have detected. That is consistent with the hypothesis that more of the same distribution buys compression rather than competence. The experiment that demonstrates the data-constrained mechanism directly—four from-scratch arms at $1\times$ /$2\times$ /$4\times$ /$8\times$ over a deliberately small shard, so that the largest arm really does run eight epochs, plus a fresh-data control arm at the same budget—is specified in §12. The practical conclusion for our own plan rests on the previous section's argument rather than this one's: the lever to pull is more unique, diverse, high-quality data, acquired rather than fabricated.

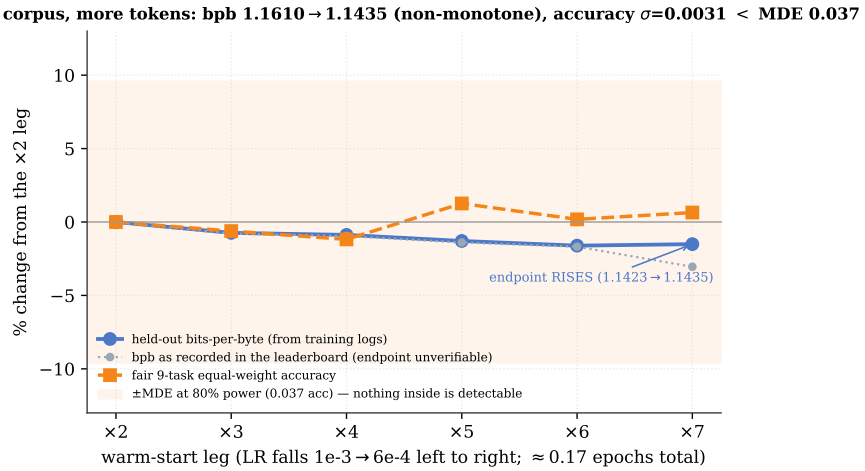

Figure 6: The token series on commensurable axes. Both curves are drawn as percent change from the first leg, so that neither is visually expanded relative to the other: scaling an accuracy axis to ±0.05 around its own mean while the bits-per-byte axis spans a 3% change manufactures a "steep versus flat" contrast out of nothing but axis choice, and this is what the same data looks like without it. Bits-per-byte is taken from the training logs (solid); the leaderboard series (dotted) is shown only to display the disagreeing final point. The shaded band is the minimum detectable effect, not a confidence interval: accuracy movement inside it is undetectable by construction. The x-axis is a warm-start chain along which the learning rate falls, and the whole series is 0.17 epochs.

Part III
New Measurements — 4 September 2026

This part adds measurements rather than arguments, and every one of them was obtained on the central processor alone, with both graphics processors committed elsewhere throughout. That constraint turned out to be informative in its own right: most of what a laboratory files under "needs a GPU" needs instead a scorer that saves its own intermediates.

7 Benchmark Contamination of Our Corpus, and of the Reference's

Contamination is measured here for the first time in this project, against a threshold fixed before the scan was run: *a published per-task rate; above 5% on any task, Table 2 is recomputed on the clean subset.* Two things make the measurement worth reporting beyond this laboratory. It

covers not only our own corpora but four components of the reference model's *published* training mixture — a comparison that is possible only because that mixture is open, and which we have not seen made elsewhere at this scale. And the detector is validated against controls before any rate from it is believed.

An evaluation-side index, not a corpus-side filter. The obvious construction — insert every corpus 13-gram into a Bloom filter — is the wrong direction by four orders of magnitude: at $\sim 6 \times 10^9$ word 13-grams it is roughly six CPU-*months* of Python, and it returns upper bounds rather than rates. We invert it. The $\sim 10^5$ *evaluation* n-grams are hashed once into a sorted array; the corpus is then streamed through a vectorised rolling hash and looked up by binary search. This makes the same measurement at ~ 1 CPU-hour per 10^{10} tokens, and it removes the Bloom filter's false positives: with 64-bit keys and $|E| \approx 10^5$ evaluation grams, the expected number of spurious hits over 10^{10} corpus grams is $|E| \cdot 10^{10}/2^{64} \approx 5 \times 10^{-5}$ items, so the rates below are measurements rather than upper bounds. This runs at ~ 1 CPU-hour per 10^{10} tokens on ordinary hardware, which is what makes scanning a reference model's published mixture practical at all. Normalisation, the choice of 13-grams and the question-plus-answer construction follow the pre-registered specification unchanged, and we report orders 8, 13 and 20 so the reader can see the sensitivity to the arbitrary standard choice.

The detector is validated before it is trusted. A contamination rate is only as good as the detector's false-negative rate, which is never reported. We therefore build both controls out of the corpus itself: 64 windows of 40 words lifted verbatim from the shard become synthetic evaluation items, and the same windows with their words permuted, plus 64 windows of words drawn uniformly from the corpus vocabulary, become negative controls. The detector finds 64/64 positives at every order and 0/64 of both negative sets. Every contamination rate below is therefore a rate from an instrument with a known false-negative and false-positive behaviour, which is the condition under which such a rate means anything at all.

Result 1: per token, the hand-curated corpus is the contaminated one

The first finding is not where anyone would have looked for it. At a *matched* scanning budget, every web corpus in this study — ours and the reference's alike — is essentially clean at the 13-gram order, and the contamination is concentrated, by more than an order of magnitude, in the one corpus in this project that was *hand-assembled for quality* (Table 4): the 2.80-million-token "curated decay" buffer that the trainer injects over the final 40% of every leg of the champion's training, at the annealed end of the schedule where the reference's own report says the mixture pays off.

What "the reference's corpus" means here. Three of the four reference-side corpora below are not proxies. They are prefixes of the corpus the reference model's own authors published [4, 1]: the Cosmopedia-v2 and FineWeb-Edu-dedup configurations of the SmolLM corpus, and the FineMath-4plus configuration of FineMath. The fourth is the DCLM baseline [22]. Two qualifications follow and both matter: we scanned a *prefix* of each, re-tokenised to our own 16k vocabulary, and the reference's actual training mixture samples from these corpora rather than consuming them whole, so a clean result on our prefix is evidence about the corpus, not proof about the particular tokens the reference saw.

Table 4: Contamination at an *identical* scanning budget: the first 2,799,576 tokens of each corpus — the full length of the curated buffer — against the same 4000 evaluation items (500 per task × 8 tasks). Counts are items containing a verbatim normalised word n-gram of the corpus.

corpus	role	8-gram hits	13-gram hits
FineWeb-10BT (our main shard)	ours	6	0
FineWeb-3B (our root run's shard)	ours	6	0
curated decay injection	**ours**	**140**	**17**
Cosmopedia-v2	reference mixture	4	0
FineWeb-Edu	reference mixture	5	0
DCLM	reference mixture	2	1
FineMath	reference mixture	7	0

At an identical budget the curated buffer carries 17 of 4000 evaluation items contaminated at the 13-gram order against 0 or 1 for every web or reference corpus, and 140 at the looser 8-gram order against 2 to 7 for every other corpus. Per task the rate reaches 1.2% (OPENBOOKQA), 1.0% (ARC_EASY), 1.0% (ARC_CHALLENGE) — all below the pre-registered 5% trigger, so Table 2 is not overturned, but all from a corpus 3266 times smaller than the shard it is injected alongside.

Why, exactly, and the mechanism it generalises to. The curated buffer is concatenated from four sources: three teacher-generated files — question–answer, reasoning and knowledge — and one multiple-choice training pool. The reasoning file is built by a generator that draws the *train* splits of ARC, OpenBookQA, CommonsenseQA, HellaSwag and GSM8K and *does* filter against the held-out splits — but the filter is an exact match on the normalised full question string, it indexes the question field only and never the passage or the answer, it covers six datasets and not BOOLQ, MMLU or WINOGRANDE, and each dataset's load is wrapped in a bare exception handler that logs a failure and continues. The multiple-choice pool — 21,068 items, no recorded provenance and no surviving builder — passes through no filter at all. Tracing the flagged items back to their sources: 15 of the 17 enter through that unfiltered pool and 5 through the filtered reasoning file. Both figures carry a general lesson. The first is that a corpus assembled from several sources inherits the *weakest* decontamination among them, so a mixture is only as clean as its least careful constituent. The second is that exact question-string matching — the filter this project believed it had, and a common choice — does not stop 13-gram leakage, which is the standard finding of the contamination literature [42, 43, 46] reproduced here on data whose construction we control end to end.

Three of the seventeen leaked spans reach into the correct answer. The flagged items are ordinary benchmark items, verbatim: ARC-Easy's *"Which of the following elements is best able to combine with itself and hydrogen [H] to form large molecules?"*, OpenBookQA's oak-tree-and-sidewalk item, and — from a teacher-generated knowledge file, not from any benchmark split — the opening sentence of the BOOLQ passage on the *Titanic*. Every flagged item is recorded in full, with the source it entered through, in this paper's committed measurement record.

Result 2: at corpus scale, nothing is clean — including the reference's

Matched budgets are the right way to compare corpora and the wrong way to state absolute risk. A 2.80-million-token window is a 3266th of our own shard, and contamination accrues: every accrual curve in Figure 7 is still rising at its right-hand edge. Scanned to their full extent the picture changes, and it changes on *both* sides.

The one reference-mixture component we were able to scan to completion within this pass is FINEMATH (3.0×10^9 tokens, the whole of our local copy of the FineMath-4plus configuration). It is not clean (Table 5).

Table 5: Contamination of FINEMATH scanned to completion (3.0×10^9 tokens), by n-gram order. Each entry is the fraction of 500 evaluation items for that task containing a verbatim normalised word n-gram of the corpus. "others" is the largest value over HELLASWAG, WINOGRANDE, OPENBOOKQA and COMMONSENSE_QA.

	MMLU	ARC_E	ARC_C	BOOLQ	others
8-gram	0.1740	0.1020	0.1280	0.1540	≤ 0.0080
13-gram	**0.0280**	0.0020	0.0060	**0.0500**	0.0000
20-gram	0.0100	0.0020	0.0040	0.0380	0.0000

BOOLQ reaches exactly the 5% figure pre-registered as the trigger for recomputing the head-to-head on a clean subset, and 3.8% of BOOLQ items survive even at the 20-gram order — not a coincidence of common phrasing, since BOOLQ passages are encyclopaedic prose and mathematics-filtered web text contains a great deal of it. MMLU is at 2.8% at 13-grams and 1.0% at 20-grams. Neither task is one this project's models can do — §9 shows MMLU is cleared above chance by two of seventy-two runs and that both of them are among our smallest models — so this is not an explanation of our own scores. It is a measurement about a corpus a widely used small model was trained on, possible only because that corpus is published, and it is the reason a decontamination statement belongs in every small-model report rather than in almost none of them.

The comparison is a like-for-like measurement of two sets of corpora with one validated detector, made on a prefix of a published corpus rather than on the tokens the reference model consumed, and with no visibility into whatever decontamination its authors applied downstream of publication.

And on the other five corpora, at a common order of magnitude. The five remaining corpora were scanned in parallel to a common budget and stopped at a chunk boundary when the shared processors ran out; each therefore reports a *prefix* at the budget it reached, and that budget is stated with every rate here and in Figure 7. At this order of magnitude the answer is that everything is mildly contaminated and nothing distinguishes our web data from the reference's.

Table 6: The five corpora scanned in parallel to a common target, reported at the prefix each reached. "8-gram" and "13-gram" are counts of contaminated items out of 4000; the last two columns are the per-task 13-gram rates for the two benchmarks that leak.

corpus	Mtok scanned	8-gram	13-gram	MMLU	BOOLQ
FineWeb-10BT (ours)	248	136	14	0.0040	0.0220
FineWeb-3B (ours)	256	139	16	0.0040	0.0260
Cosmopedia-v2	248	81	4	0.0020	0.0060
FineWeb-Edu	288	144	15	0.0080	0.0220
DCLM	280	85	6	0.0060	0.0060

Our two shards return 14 and 16 contaminated items of 4000; the reference's three return 4, 15 and 6. There is no differential here worth a sentence of interpretation, which is itself the useful finding: the *gap* in Table 2 cannot be explained by one side having seen more of the benchmark than the other. What survives is the per-token result of Table 4 — our own hand-curated buffer — and the absolute level in Table 5, which is a caution to everybody rather than a differential between two models.

What this does and does not license. The pre-registered trigger — "if any task exceeds 5%, Table 2 is recomputed on the decontaminated subset and both versions are published" —

was written against *our own* pretraining shard. On the corpora our champion actually trained on, no task reaches it. We nevertheless publish both versions of the comparison (item 1 of §12), because the trigger is a floor on disclosure and not a ceiling, and because FINEMATH reaching 5.0% on BOOLQ shows the threshold is not a comfortable distance away for anyone.

What the measurement does establish is a mechanism the "curate a high-quality annealing mixture" recipe of §11 walks straight into. The corpus a laboratory hand-assembles for quality is the corpus most likely to be built out of benchmark-shaped material — because benchmark-shaped material is exactly what a quality filter selects for — injected at the end of training where its effect on the benchmark is largest, and filtered by whatever decontamination its weakest constituent happened to have. Per token, our web-scale data was clean and our careful data was not. That is the opposite of the intuition the plan was built on, and it is the reason item 11 of the queued work is a decontaminated retrain rather than a further measurement.

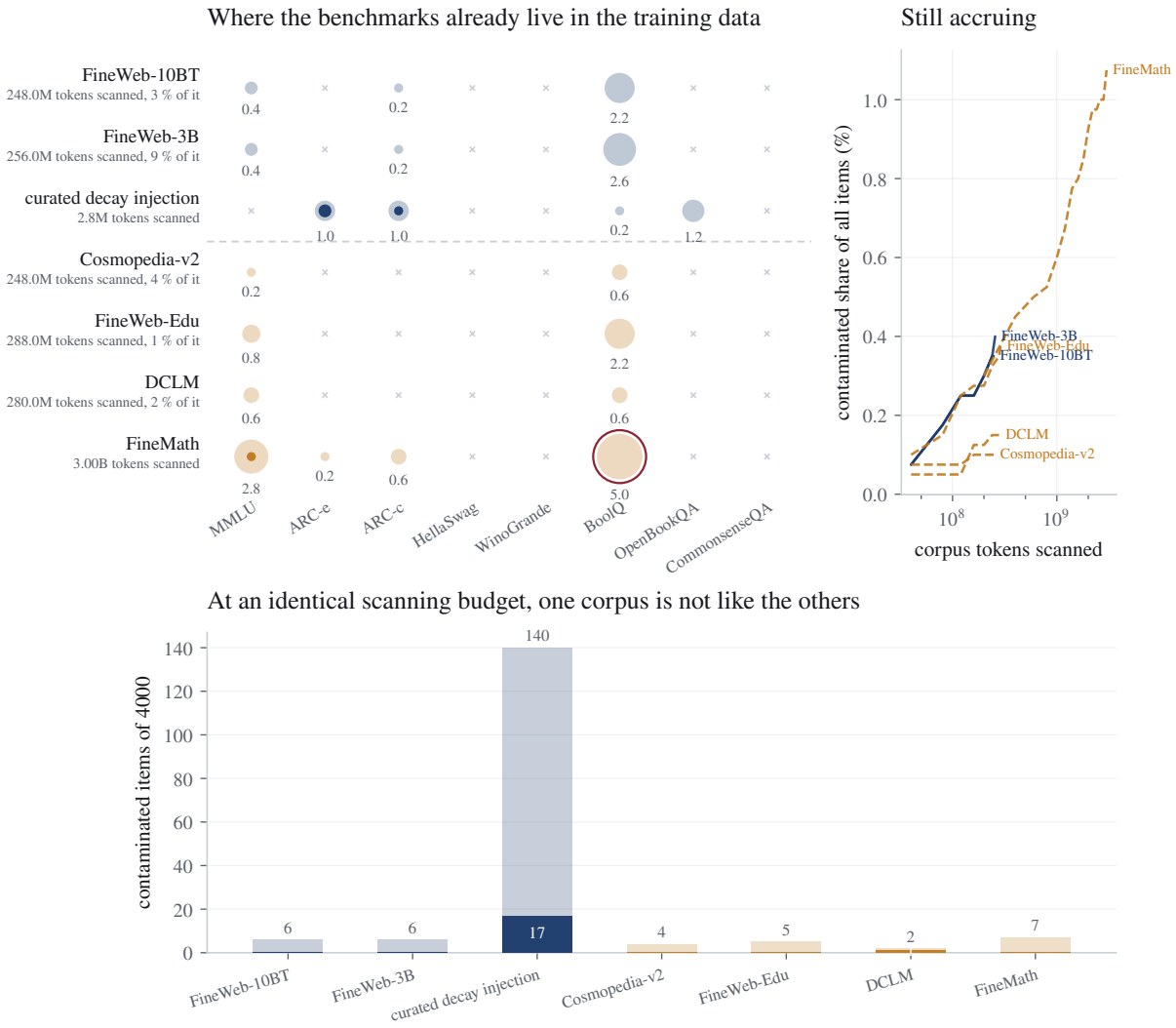

Figure 7: Verbatim benchmark overlap of every corpus this project trains on and of four components of the reference model's published mixture. Upper block: our own pretraining corpora. Lower block, below the dashed rule: components of the reference's published mixture. *Top left,* where the benchmarks already live in the training data. Disc area is the share of evaluation items whose question-plus-answer contains a verbatim 13-gram of the corpus, printed beneath each disc as a percentage; the solid core is the share whose matched span reaches into the correct answer; an oxblood ring marks a rate above the 5% threshold fixed before the scan. A cross means no overlap at that order. The row label gives the budget each corpus was scanned to and what fraction of it that is. *Top right,* still accruing: contaminated share against corpus tokens scanned. Every curve rises at its right-hand edge, so these rates are lower bounds on what a complete scan would find. *Bottom,* at an identical scanning budget, one corpus is not like the others. Every bar is the same 2.80M tokens of each corpus against the same 4000 evaluation items; the pale bar is 8-gram overlap and the solid bar 13-gram. The corpus injected into the final 40% of every leg of the champion's training is the only one with material 13-gram overlap at all — it is built from benchmark train splits, formatted to match the evaluation prompt, and filtered only by exact question-string match. A representative flagged item: ARC-Easy's *"Which of the following elements is best able to combine with itself and hydrogen [H] to form large molecules?"*, whose correct answer "carbon [C]" is present verbatim in the unfiltered multiple-choice pool.

8 How Fragile Is a Composite? Leave-One-Task-Out

Section 4 localised the learning-rate effect by deleting BOOLQ and observing that it nearly vanished. A referee's next question is the obvious one: *is* BOOLQ *special, or would deleting any single task have done that?* A composite over eight tasks in which *any* deletion halves the effect is simply an unstable statistic, and the conclusion would then be about the metric's variance rather than about one benchmark's class prior. This is the test that distinguishes the two, and it is cheap enough that every composite-based result deserves it.

We therefore recompute the whole leave-one-task-out (LOTO) profile from the sweep's own per-example margins, with a paired bootstrap ($B = 5000$, resampled within task so the pairing across arms is preserved) at every deletion (Table 7, Figure 8).

Table 7: Leave-one-task-out on the learning-rate comparison 1.5×10^{-3} versus the prior default 6×10^{-4}, over the sweep's own per-example margins ($n = 1000$ per task, identical items across arms). "Ratio" is $M(1.5\times10^{-3})/M(6\times10^{-4})$ on the composite that remains after the named task is deleted.

task deleted	ratio	Δ	95% CI	p
— (all eight tasks)	2.93×	+0.1342	[+0.0998, +0.1671]	< 0.001
MMLU	3.29×	+0.1535	[+0.1159, +0.1909]	< 0.001
ARC_EASY	4.04×	+0.1512	[+0.1130, +0.1896]	< 0.001
ARC_CHALLENGE	2.49×	+0.1465	[+0.1088, +0.1845]	< 0.001
HELLASWAG	3.72×	+0.1503	[+0.1105, +0.1895]	< 0.001
WINOGRANDE	2.91×	+0.1513	[+0.1119, +0.1897]	< 0.001
BOOLQ	**1.16×**	**+0.0195**	[**−0.0017**, **+0.0409**]	**0.074**
OPENBOOKQA	3.71×	+0.1526	[+0.1161, +0.1890]	< 0.001
COMMONSENSE_QA	6.26×	+0.1488	[+0.1128, +0.1862]	< 0.001

The answer is unambiguous. Deleting BOOLQ takes the effect from $\Delta = +0.1342$ ($p < 0.001$) to +0.0195 ($p = 0.074$) and the ratio from 2.93× to 1.16×. Deleting *any other* single task leaves the effect between +0.1465 and +0.1535, all with $p < 0.001$, and leaves the ratio between 2.49× and 6.26×. The result of §4 is therefore not an artefact of an unstable metric: it is one benchmark, and the composite is otherwise robust to deletion.

What "one task carried it" means at the level of examples. A mean can be moved two ways: a task's whole distribution can translate, or one task's margins can simply live on a wider scale than the others' and so dominate an unweighted average of per-task means. The second is the defect Eq. (1) diagnoses, and it is the obvious suspect here, because BOOLQ's answers are the strings "no" and "yes" — a mean of 2.63 bytes, against 9.98 to 60.66 for every other task — and the scorer divides by exactly that byte length. We tested it rather than asserting it, and *it is not what happened*: BOOLQ's per-example margin standard deviation is 1.92, fourth of eight and close to the median of 1.77, while the widest task is COMMONSENSE_QA at 2.46 and the narrowest HELLASWAG at 0.91. BOOLQ does not dominate by being wider.

It dominates by *moving*. Between the prior default and the arm that was crowned, its mean per-example margin translates by +0.9375; the second-largest movement over the other seven tasks is +0.0481, a factor of 19 smaller (Figure 9). Seven of the eight distributions are visually indistinguishable between the two arms. An unweighted mean of eight per-task means is maximally sensitive to exactly this: one member translating bodily while the rest stand still. That is the whole of the 2.93×, drawn at the resolution of individual examples.

The same test, applied to the result we keep. The identical procedure runs against the headline we retain — the seven-task smooth-margin lead of our model over the reference —

in item 1 of §12, where a paired measurement of both models on identical items makes the paired bootstrap legitimate. A fragility test applied only to the results one has already set aside automates the wrong half of the procedure.

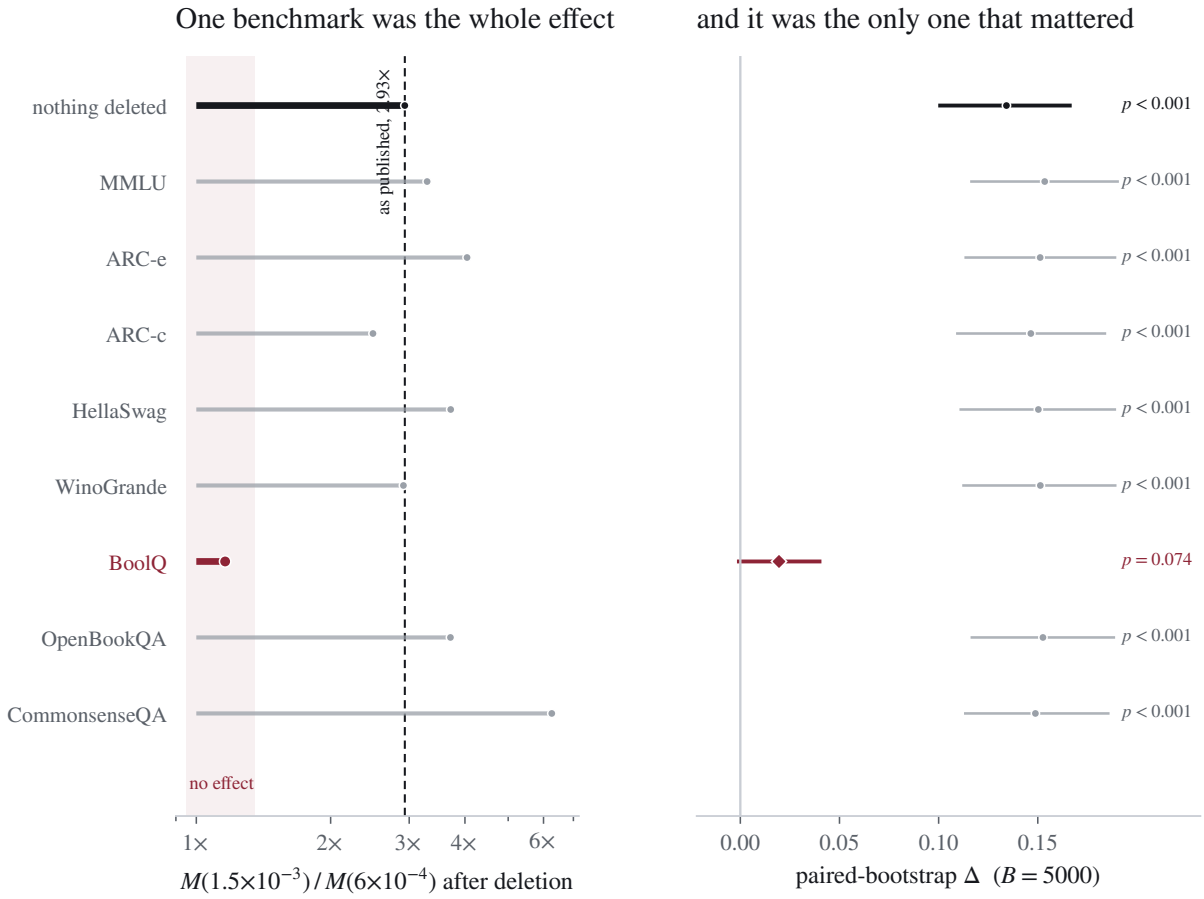

Figure 8: Leave-one-task-out on the eight-task composite that carried the learning-rate effect. *Left:* the ratio $M(1.5{\times}10^{-3})/M(6{\times}10^{-4})$ surviving each single-task deletion, against the eight-task value of 2.93× (dashed vertical). The shaded band at the left is the no-effect region, and only the BOOLQ deletion lands in it. *Right:* the paired bootstrap ($B = 5000$, $n = 1000$ per task) at each deletion. Every interval excludes zero except the one that deletes BOOLQ: deleting it moves the effect from $+0.1342$ ($p < 0.001$) to $+0.0195$ ($p = 0.074$), while deleting any other single task leaves it between $+0.1465$ and $+0.1535$. BOOLQ's own margin by learning-rate arm is -0.29, $+0.64$, -0.26, -0.58 at $6{\times}10^{-4}$, $1.5{\times}10^{-3}$, $2.5{\times}10^{-3}$ and $4{\times}10^{-3}$ respectively — a lone excursion at exactly the arm the aggregate crowned, and that is where the effect lived.

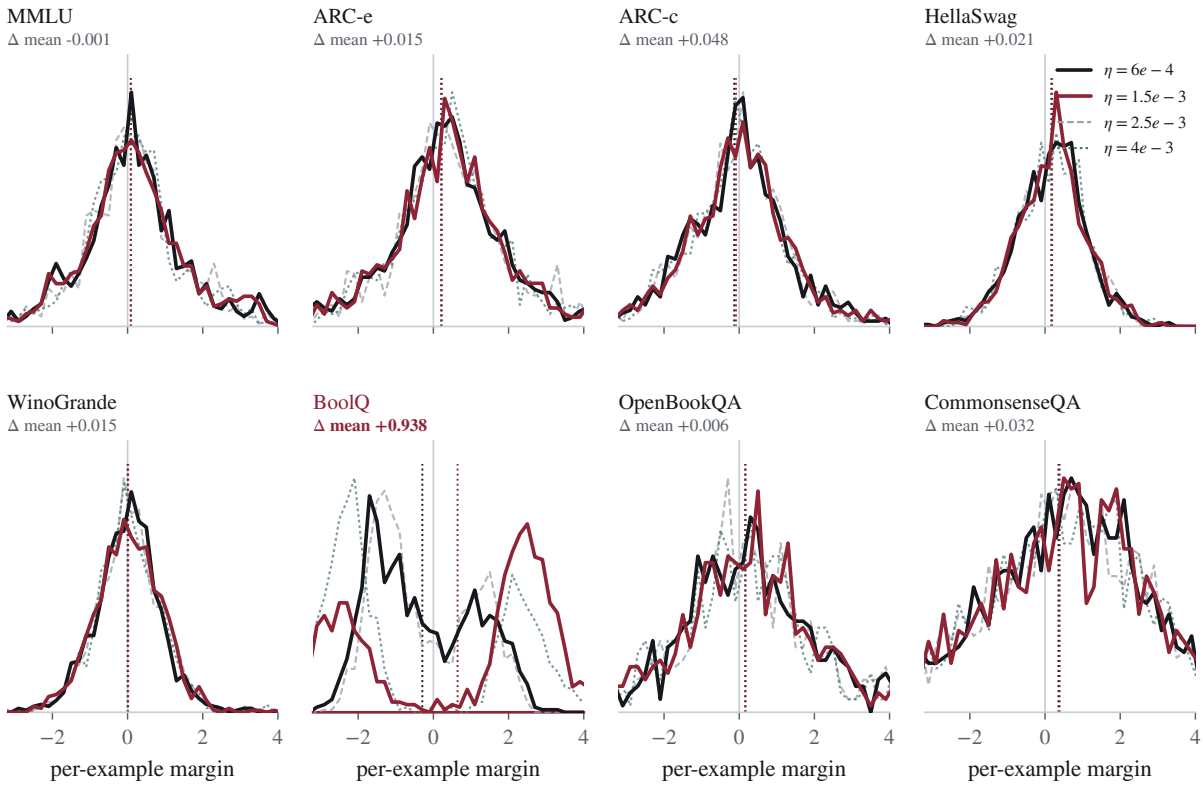

Figure 9: A composite moved by one benchmark, seen at the level of single examples. Per-example byte-normalised margin distributions for all four learning-rate arms, $n = 1000$ items per task, the same items in every arm; dotted verticals mark each arm's mean and the panel header gives the mean shift between the long-standing default (black) and the arm the aggregate crowned (oxblood). Seven of the eight panels are indistinguishable between those two arms. Only BOOLQ's translates, by +0.9375 against a second-largest movement of +0.0481 elsewhere — and the unweighted eight-task mean of these panels is what produces the 2.93×. BOOLQ's margin *spread* is fourth of eight and close to the median, so it does not dominate by being wider: it dominates by moving.

9 Seventy-Six Runs: Forecast Error and Per-Task Emergence

This project holds seventy-six evaluated checkpoints spanning 10.3 to 158.6M parameters, and until now it had extracted from them exactly one scalar comparison. Fitting $L(N, D)$ over that record would turn the plan of §11 from a hope into a forecast with error bars, so we fit it. The fit misses its own pre-registered held-out bar, and the measured forecast error is a more useful number to publish than the forecast would have been — as is the second result the same ledger yields, which bears directly on the emergence debate.

Recovering the covariates. The leaderboard carries N and bits-per-byte but no token count, no learning rate, no corpus and no warm-start ancestry, which is why no such analysis had ever been attempted. All of it is recoverable from the argument and completion lines of the 313 training logs. Walking each run's warm-start pointer back to a from-scratch root resolves a cumulative token budget for 72 of the 76 rows; the other four have no parseable argument line and are excluded rather than imputed.

Our own token budget, re-measured from each leg's arguments. A cumulative budget computed from a project-default assumption — batch 16, sequence length 512 on every leg — is an estimate, not a measurement, and reading each leg's own arguments is what turns it into one. Seven of the eight legs match the assumption. The from-scratch root does not: it was launched at sequence length 1024 and batch 16, crashed 109 seconds in, and *relaunched at sequence length* 512 *and batch* 8. The step count in its completion line belongs to the surviving launch, so its contribution is $54{,}513 \times 8 \times 512 = 2.23 \times 10^8$ tokens rather than the 4.47×10^8 the default implies (Table 8).

Table 8: Our training budget, re-measured from each leg's own arguments rather than from a project-default assumption. The re-measurement *lowers* our own budget, which makes the comparison to the reference less favourable to us, not more.

	project default (Table 1)	re-measured	source of the difference
tokens seen	1.967×10^9	$\mathbf{1.744 \times 10^9}$	the root leg ran batch 8, not 16
tokens / parameter	12.4	**11.0**	
fraction of Chinchilla	0.62	**0.55**	
reference / ours	1017×	**1147**×	

Both figures are of the same order and neither changes any conclusion. The re-measurement is reported because it is a worked instance of the rule this paper argues for: a number derived from an assumption about a file is not a measurement until it is read off the file, and the two differ here by 11% on the denominator of the paper's central comparison.

Bits-per-byte is not a common ruler either. Before any fit is possible one has to ask what the 76 bits-per-byte values have in common, and the answer is: less than one would like. The trainer evaluates held-out bits-per-byte on the final twentieth of *whichever corpus the run trained on*. Runs on different corpora therefore report compression on different distributions, and the column is comparable only within a corpus family. Of the 72 resolvable runs the largest single family has 31 members; a further 14 are on two other FineWeb variants; 14 are on code or bytecode corpora whose bits-per-byte is not commensurable with web text at all (its minimum over the record is 0.290, against 1.126 for web); and 13 do not record which corpus they trained on. This is the same class of defect as the token-count normalization repaired in §3, one level up: an aggregate whose elements were never on one scale.

The fit, and the held-out test it has to pass. On the largest family ($n = 31$, N from 15.4 to 158.6M, D from 3.4×10^7 to 1.9×10^9) we fit Eq. (5),

$$L(N, D) \;=\; E \;+\; \frac{A}{N^{\alpha}} \;+\; \frac{B}{D^{\beta}} \tag{5}$$

in the standard way — residuals in log space under a robust loss, multi-start over the exponents — obtaining $E = 1.213$, $\alpha = 1.76$, $\beta = 1.93$. The queued specification asked for a held-out prediction within 2%. Two legitimate held-out tests were fixed before the fit was run, because the champion trains on a *different* corpus from this family and predicting it would not be a fair test:

1. **Leave-one-out** over the family: median relative error **4.5**%, worst 64.8%.
2. **Extrapolation**, the test the plan actually needs: refit with the top quartile of D withheld, then predict it. Median relative error **5.8**%.

Both miss the 2% bar by more than a factor of two, so **the fit is reported as a bound on forecast error rather than as a forecast**. The practical consequence is worth stating

numerically: the two fits above — the same functional form on nested subsets of the same data — predict, at the 10^{11}-token budget of §11, bits-per-byte of 1.213 and 1.181. A spread of 0.032 between two fits of one dataset is a quarter of the entire improvement (1.269 → 1.142, i.e. 0.127) that the champion's whole 1.7-billion-token lineage produced — so a forecast from this record carries an error bar comparable to a quarter of everything the lineage achieved. This record cannot forecast this laboratory's next run, and the useful transferable number is that bound on the forecast error rather than the point prediction it brackets. What would move it is stated in §12: seeded replicates, and a family fitted on one corpus at a time.

Emergence: there is no threshold to find. The same ledger answers a question the paper cites but had never informed. For each task and each run we ask whether the run's Wilson 95% lower bound clears the task's own chance level, and we control the eight-way multiplicity with Benjamini–Hochberg at $q = 0.05$. The result (Table 9, Figure 10) is not a set of thresholds. It is a dichotomy:

Table 9: Per-task chance-clearing across the 72 runs with a resolvable budget. "First" is the smallest-budget run whose Wilson 95% lower bound exceeds chance.

task	chance	runs clearing	first clears at N	at D	survives BH
ARC_EASY	0.25	70/72	15.4M	3.4×10^7	yes
HELLASWAG	0.25	70/72	15.4M	3.4×10^7	yes
COMMONSENSE_QA	0.20	56/72	158.6M	1.0×10^8	yes
BOOLQ	0.50	52/72	158.6M	1.0×10^8	yes
OPENBOOKQA	0.25	22/72	158.6M	1.0×10^8	yes
MMLU	0.25	2/72	10.3M	4.8×10^8	yes
ARC_CHALLENGE	0.25	**0/72**	—	—	no
WINOGRANDE	0.50	**0/72**	—	—	no

1. Two tasks — ARC_EASY and HELLASWAG — are above chance for the *smallest-budget run in the entire record* — 15.4M parameters and 3.4×10^7 tokens, the minimum D over all 72 resolvable runs. There is no threshold to locate for these two: they are cleared from the very first run onward.
2. Two tasks — ARC_CHALLENGE and WINOGRANDE — are cleared by *no run at all*. Seventy-six trained models, a fifteen-fold parameter range and a fifty-seven-fold token range have never put either above chance. Both appear in Table 2; on ARC_CHALLENGE that table records our model *leading*, which this measurement reframes as a difference between two models that are both at chance.
3. MMLU is cleared by two runs out of seventy-two, and both of them are among the *smallest* models this project has ever trained, at 15.4M and 10.3M parameters. Our 158.6M champion does not clear it. A benchmark whose only two successes across a fifteen-fold parameter range come from the two smallest models in the range is measuring sampling noise, not knowledge — and it is the benchmark on which Table 2 records the reference's second-largest lead.

The reachable range here is far below where the emergence debate is conducted, and these are single runs without seed replicates. What the ledger supplies is a receipt-backed data point of the kind that debate asks for and rarely gets: over a 15× parameter range on one rig, no per-task capability threshold was observed — tasks were either cleared from the smallest model or never cleared at all.

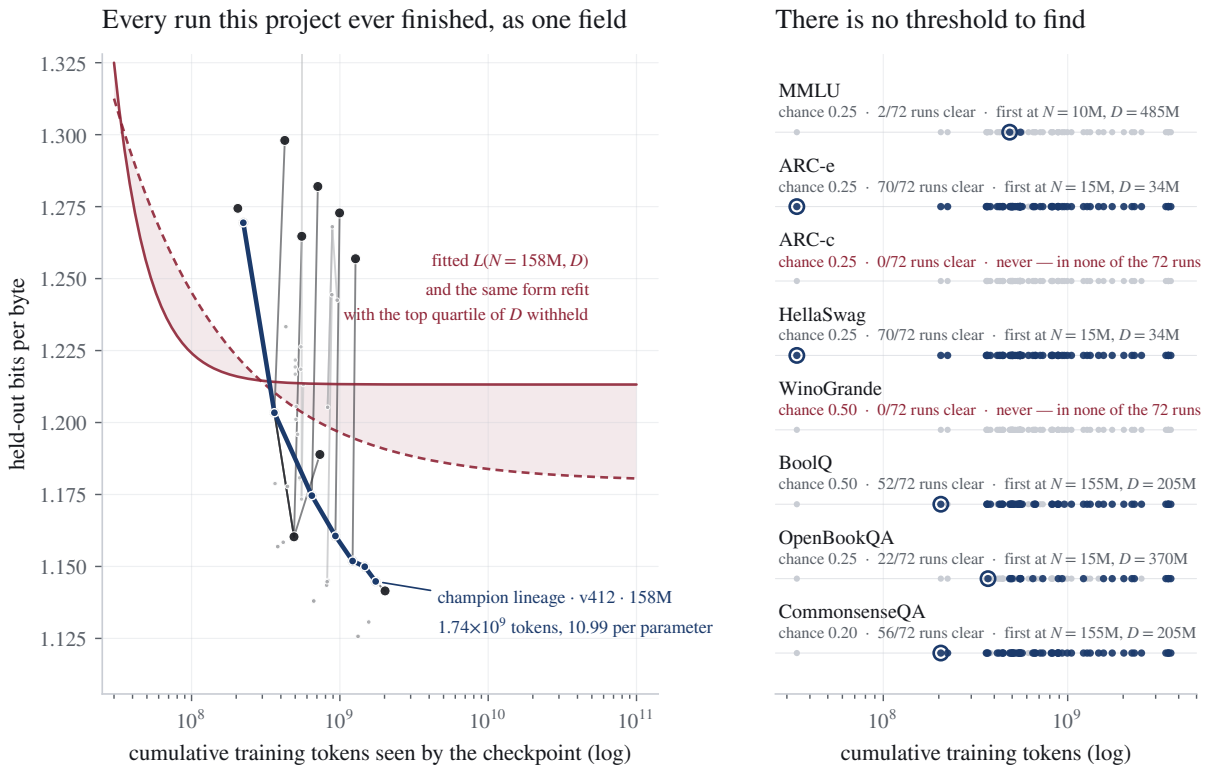

Figure 10: *Left:* every finished run on a FineWeb corpus (45 of the 76), drawn as warm-start trajectories in (cumulative tokens, held-out bits-per-byte); the champion lineage in blue. Red: Eq. (5) fitted on the 31-run single-corpus family, and the same form refit with the top quartile of D withheld; the shaded band between them is the forecast error and it is a quarter of the entire improvement the champion lineage achieved. *Right:* per-task chance-clearing across the 72 runs with a resolvable budget, with the multiplicity-controlled verdict; two of the seven headline tasks are cleared by no model this project has ever trained, and MMLU is first cleared by its smallest model of all. One further web-corpus run lies above the plotted range and 22 runs are excluded because they trained on a code or bytecode corpus or their log does not record which corpus: bits-per-byte is measured on each run's own held-out tail, so it is not comparable across corpora.

Part IV
Method and Plan

10 Adversarial Multi-Agent Auditing

The method that produced the results above is worth stating on its own, because it is transferable and because it is not the method one would expect. None of them came from a training run. Each came from convening a small panel of independent analytical agents, charged with proposing a diagnosis and then adversarially auditing one another's diagnoses against the saved record of results. That process is what surfaced the class-prior artifact, the scale-domination of the composite, and the download-versus-manufacture decision — three results, zero GPU-hours.

It also works on the write-up. On 2 September 2026 the same procedure was pointed at the manuscript itself rather than at the experiments, with one instruction: trace every number to the artefact that produces it. It returned twenty findings, of which the three sharpest are the

substance of §4, §7 and Table 2 — an aggregate effect that was one benchmark, a comparison whose two columns had to be put on one ruler before they meant anything, and a scaling law cited for a regime the runs never entered. Not one of them required a graphics processor to find; all three were found by arithmetic on files that had been sitting in the repository the whole time. That asymmetry is the practical point, and it is the reason to run this procedure at all: the audit that redirects your research programme is almost always cheaper than the experiment that would have redirected it.

One number, one artefact. Every number in this paper is emitted by a committed analysis script that reads only committed artefacts, runs on the central processor, and writes one machine-readable record per result: the learning-rate sweep with its paired bootstrap, the head-to-head table with per-task confidence intervals, the pinned leaderboard statistics, the composite decomposition, the token-series provenance and power analysis, our own training budget, and the harness-version sensitivity of fixed checkpoints. The figure generators read those records and a frozen leaderboard snapshot; they contain no benchmark literals. The whole paper is reproducible on a machine with no GPU at all.

Results as regression tests. A second executable test pins each stated result to the artefact that produces it and to the class of stronger statement the data does not reach: the learning-rate ratio is 1.16 ex-BOOLQ at $p = 0.074$, the reference leads five of seven at a sign test of 0.45, the budget gap is of order 10^3, no run exceeds 0.0833 on GSM8K or 0.02 on MBPP and none is non-zero on HumanEval. Fifty-one such checks run on every build, each matching a class of wording rather than one phrasing, and each negative-tested against a synthetic sentence that states the claim it exists to catch. A laboratory's assertions are version-controlled by convention; the numbers that size them usually are not, and they are what erode.

11 The Plan the Measurements Support

The plan is no longer a guess, because the reference's technical report supplies the recipe we must match. The reference model was trained on the order of two trillion tokens, three orders of magnitude above our budget. Its report is explicit that, at our scale, the lever is not tokens but the *mixture*: its small models use a single-stage schedule of consistently high-quality, freely downloadable corpora—a roughly sixty-forty blend of educational and filtered web text, an educational-code slice, and, above all, a step-by-step mathematics corpus that the report credits with doubling grade-school-math accuracy. Every one of these is downloadable, which is §5 made concrete. Our plan is therefore to reproduce the reference architecture to the parameter, train on a feasible budget of some one hundred billion unique downloaded tokens—on the order of a thousand tokens per parameter, a run of some two processor-weeks, and roughly fifty times our entire cumulative budget to date (Table 1)—over this same mixture, and to grade the result against the reference *on our own harness*, the only comparison this work has shown to be trustworthy. The objective is to erase a measured gap of 2.8 points of mean accuracy.

Each lever in hand is sized separately. The data lever is: the corpora are identified, downloadable, and unambiguously larger and more diverse than what we have trained on, and §7 adds a requirement the plan did not previously have — the annealing mixture must be decontaminated *before* it is built, not after. The learning-rate lever is not yet sized: §4 bounds it well below the threefold figure the eight-task aggregate suggested, and the seeded, token-matched sweep will fix its magnitude; what is known independently is that the reference chose a value five times hotter than our default. The token budget is a lever of known partial reach: 10^{11} tokens is 5% of the reference's, so a flawless execution of this plan closes part of a 1017× data-scale deficit, and the iso-token baseline of §12 is what quantifies the remainder. Stating the reach of each lever separately is what makes the plan executable rather than aspirational.

12 Open Problems and the Research Programme They Define

The measurements above settle several questions and sharpen the ones they do not. This section states the sharpened ones as a programme: each is an experiment with a runnable specification, a cost, and a success criterion fixed *before* it runs, so that its outcome is informative in either direction. Committing the criterion in advance is what makes a negative result publishable rather than discardable, and every item here is written to that standard. Ordered by what they would buy, not by what they cost.

The five that would extend the central comparison

1. **The head-to-head at matched n, matched precision and matched device** ($\approx$ 2 GPU-h, or about two CPU-hours per model on an idle machine — the scorer that makes it possible now persists per-choice log-likelihoods, byte lengths and token counts, which is what converts this from a GPU experiment into ordinary processor work). Both models at $n = 1200$ in `float32` on one device, paired bootstrap on the margin and exact McNemar on per-item correctness, with per-class and balanced accuracy recorded throughout, and the same comparison repeated on the decontaminated subset of §7. *Criterion:* if the paired interval on the seven-task accuracy gap excludes zero, the lead in Table 2 stands as a paired measurement; if it includes zero, the table is restated as no detected difference at this n. Because the pairing makes a per-item test legitimate, this is also where the leave-one-task-out procedure of §8 is applied to the margin lead we retain.

2. **Does recalibration close the gap?** ($\approx$ 0.5 GPU-h, and CPU-feasible on the same per-choice quantities.) The decision rule is fitted rather than assumed: for each choice i with raw summed log-likelihood S_i, byte length B_i and token count T_i, the shipped rule $\arg\max_i S_i/B_i$ is generalised to $z_i = (S_i + aT_i + cB_i)/B_i^b$ applied *before* the argmax, fitted per task on one half of the items and evaluated on the other, both directions. The shipped scorer is the point $b = 1$, $a = c = 0$, so the family strictly contains it. The identical family is fitted to the reference, because a claim that *our* model is merely miscalibrated requires that recalibration help us more than it helps them. *Criterion:* closing $\geq 50\%$ of the accuracy gap establishes "knowledge close, calibration lagging"; below 10% refutes it and the phrase leaves the paper; between the two, the number is reported as measured.

3. **An iso-token, iso-parameter public baseline** ($\approx$ 4 GPU-h). This is the highest-value experiment in the programme, because it separates the two hypotheses the whole paper turns on. Pythia-160M is a parameter-exact match to our champion and publishes intermediate checkpoints; the one near 2×10^9 tokens matches our own budget almost exactly. *Criterion:* if our model matches or beats it, the recipe is competitive at matched tokens and the residual gap to the reference is quantified as a data-scale gap — the strongest positive claim available to a laboratory at this budget. If it does not, the residual is recipe, and §1 is restated accordingly. Either outcome is a result; this is the single experiment that converts the paper's premise into a measurement.

4. **A seeded, token-matched learning-rate sweep** ($\approx$ 11 GPU-h). Five rates, three seeds each, a step cap rather than a wall clock, graded on ex-BOOLQ margin *and* bits-per-byte. *Criterion:* an arm must beat the long-standing default by more than 2σ of the seed spread *and* not be worse in bits-per-byte. Beyond settling §4, this run produces the first run-to-run noise floor σ this project has ever had, and every Δ in this paper — and in the programme below — gains a scale to be read against. It is the cheapest single purchase of statistical power available here.

5. **A decontaminated retrain** ($\approx$ 25 GPU-h). §7 establishes an association: the corpus injected at the annealed end of training is the one carrying benchmark n-grams. Rebuilding

that buffer with 13-gram decontamination against every held-out split, and repeating the champion's final leg from the same checkpoint, turns the association into a causal estimate. *Criterion:* the per-task accuracy difference between the two legs, with a paired interval, is published whichever way it points — a null is as informative as a drop, because it bounds how much benchmark-shaped annealing data is worth.

Extending the contamination measurement

The scan in §7 is, to our knowledge, the first published n-gram contamination measurement of a widely used small model's *own* training mixture, and it is bounded by the budget it reached rather than by anything intrinsic. Three extensions follow directly and each is CPU-only. Scanning the remaining mixture components to completion converts every prefix rate in Table 6 from a lower bound into a measurement; the accrual curves in Figure 7 give the expected direction, so the pre-registered question is how far each rises rather than whether it does. Extending the detector to near-duplicate matching — suffix-automaton or MinHash — measures the paraphrase leakage that verbatim n-grams cannot see, against the same validated controls. And publishing the detector against other open mixtures would let the field report contamination as a routine number. *Criterion, fixed in advance for all three:* rates are published at whatever budget is reached, with the budget stated, and any task exceeding 5% triggers a clean-subset recomputation of the affected comparison.

Sharpening the instrument

6. **Balanced accuracy across the suite**, extending item 1's per-class breakdown to every benchmark rather than to BOOLQ alone. *Criterion:* BA in $[0.48, 0.52]$ substantiates the class-prior reading of §3 for a task; above 0.55 refutes it for that task. Applied suite-wide this yields a per-task prior-sensitivity profile, which is a more useful object than a single verdict.

7. **A matched generative table** ($\approx$ 5 GPU-h): HumanEval and MBPP PASS@1/@10 and GSM8K exact match for our model *and* both baselines, zeros included. §3 states our own scores; what this adds is the reference's, on the same harness, which is the only way to know whether generation at this scale is a model property or a scale property.

8. **A genuinely controlled token series** ($\approx$ 25 GPU-h): four from-scratch arms at $1\times/2\times/4\times/8\times$ over a shard small enough that the largest arm truly runs eight epochs, at fixed learning rate, without distillation or corpus injection, plus a fresh-data control arm at the same budget. *Criterion:* the repeated-data and fresh-data arms must diverge by more than 2σ of the seed spread for the data-constrained mechanism to be demonstrated on this rig. §6 explains why the existing series cannot answer this and this design can.

9. **Re-evaluation of the final leg's bits-per-byte** ($\approx$ 0.3 GPU-h), resolving the 0.018 disagreement between its training log and the leaderboard. Small, and it closes the one open provenance question in Figure 6.

10. **Seeded replicates for the scaling fit.** §9 reports a forecast error of 4.5% leave-one-out and 5.8% extrapolated against a 2% bar. The record it is fitted on has one run per configuration and mixes corpora; refitting per corpus family with seed replicates is what would establish whether the 2% bar is reachable on a rig of this size at all — a question the scaling literature does not answer for laboratories at this scale, and one this ledger is unusually well placed to ask.

Items 1–3 are the ones on which the present paper's central comparison depends, and they total under forty GPU-hours — roughly a day and a half of the rig, and the first two are CPU-feasible.

Four results are already in hand: an evaluation instrument rebuilt so that a cross-vocabulary comparison is legitimate, with three separate defects in it recovered by arithmetic alone; a learning-rate effect localised to a single class-prior benchmark at the resolution of individual examples, with the alternative mechanism measured and refuted; a contamination measurement of our own corpora and of a reference model's published mixture, with a validated detector, whose central finding inverts the intuition the field's annealing recipes are built on; and a seventy-six-run ledger that yields both a measured bound on forecast error and a receipt-backed data point on per-task emergence over a fifteenfold parameter range.

13 Related Work

This study sits at the intersection of several mature literatures; our contribution is not to found any of them but to expose, on a fixed rig, how their known results interact—and how neglecting them silently taxed every experiment we ran. We organize the prior art into the threads that each of our results touches.

Small models and the data-centric turn. The premise that a 100–500M model can capture a large fraction of frontier competence is by now well established. Our reference, SmolLM2 [1], is the direct exemplar, and its lineage runs through the "textbooks" program—phi-1 [6] and phi-1.5 [7]—and through TinyStories [8], which showed that curated, synthetic data lets sub-10M models produce coherent text. Cosmopedia [4] is the open reproduction of that synthetic-textbook recipe. Architectural and training-suite work on the same scale band—MobileLLM [10], the controlled Pythia suite [9], and the Chinchilla-scaled Cerebras-GPT family [11]—frames our own 158M model as a point in a well-charted design space. The common thread, which §6 reaches independently from our own token series, is that data quality and mixture, not parameter count, separate these models.

Compute-optimal scaling and over-training. The scaling laws of Kaplan et al. [12] and the compute-optimal correction of Chinchilla [13] fix the tokens-per-parameter ratio for a *single* training pass on *fresh* data. Two refinements bear on our memorization result (Figure 6). First, small deployable models are deliberately trained far past the Chinchilla point, a regime whose predictability Gadre et al. [14] characterize as reliable over-training. Second, Muennighoff et al. [3] show that when data is *repeated* rather than fresh, the marginal value of additional epochs decays toward zero after a handful of passes. That mechanism is the natural one to reach for in reading Figure 6 and it does not apply here: our series drew 1.58×10^9 tokens from a 9.14×10^9-token corpus — 0.17 epochs — so it never entered the repeated-data regime the result describes, and no token was ever seen twice. We cite the work for what it *will* govern, namely the deliberately small-shard, multi-epoch series specified in §12, and not as an explanation of anything measured here. The regime our series was actually in is under-training, at 12.4 tokens per parameter—*below* the Chinchilla point rather than past it—which makes the reliable-over-training analysis of Gadre et al. [14] the more apposite reference for where our models sit.

Learning-rate and schedule science. The peak-learning-rate result of §4—an inverted-U in a downstream aggregate that does not survive deletion of one class-prior task, against a training objective that orders the arms the other way—is elementary in isolation but connects to an active thread, and the thread explains why a single number cannot be carried between rigs. The warmup-stable-decay (WSD) schedule introduced with MiniCPM [15], and the constant-plus-cooldown analysis of Hägele et al. [16], decouple the decay phase from a fixed horizon and thereby rationalize the warm-start continuation we pursue. The claim that our optimal rate transfers from a small-scale proxy is precisely the maximal-update-parametrization (μTransfer)

result of Yang et al. [17]. That our runs use the Muon optimizer [5]—whose scalability at LLM scale was subsequently established by Liu et al. [18]—matters because the optimal peak rate is optimizer- and batch-size-dependent; the gradient-noise-scale model of McCandlish et al. [19] and the learning-rate/batch-size interchange of Smith et al. [20] are why a single mis-set rate cannot be read off another lab's report without re-tuning.

Data curation, filtering, and mixture. The corpora our surviving plan (§11) targets are products of a large curation literature: FineWeb and its educational filter FineWeb-Edu [21], the DataComp-LM benchmark and DCLM-Baseline [22], and the classifier-filtered lineage from The Pile [23] through RefinedWeb [24] to Dolma [25]; the code slice descends from StarCoder2 and The Stack v2 [26]. Complementary techniques explain *why* these outperform raw scale: deduplication [27], importance-resampling selection (DSIR) [28], and learned mixture weights (DoReMi) [29]. Crucially, the SmolLM2 report [1] and the domain-upsampling study of Blakeney et al. [30] show that a late, high-quality *annealing* phase—the decay stage of WSD [15]—is where mathematics and educational data pay off.

Knowledge distillation as an alternative to curation. The distillation route we considered and rejected on feasibility grounds (§5) rests on classical work: logit distillation [31], sequence-level distillation [32], and on-policy LLM distillation [33]. The synthetic-textbook recipes [6, 4] are effectively offline distillation from a strong teacher. Our audit's conclusion was scope-limited rather than universal: our *local* teacher was too weak, and too slow, for that route to beat downloading the curated corpora outright.

Benchmark contamination. Decontaminating a web-scale pretraining corpus against the evaluation suite has been standard since GPT-3 [40], whose 13-gram overlap filter is the direct ancestor of the scan reported in §7; Dodge et al. [41] documented how much benchmark text a cleaned Common Crawl derivative still contains. The recent literature is largely a catalogue of the ways decontamination fails: Sainz et al. [42] argue that a contamination measurement must be published *per benchmark* rather than asserted once; Deng et al. [43] and the survey of Xu et al. [46] find leakage in modern suites that exact-match filters missed; and Golchin and Surdeanu [44] and Oren et al. [45] develop tests for the case where the corpus is not available at all. Our contribution is the measurement this literature keeps asking for and rarely gets: the corpus *is* available, so the measurement is direct, and it is run on both sides — our own shards and four components of the reference model's published mixture — at a matched scanning budget. The finding that the leakage is concentrated in the hand-curated annealing buffer rather than in the web data is, as far as we know, not something the contamination literature has reported, and it is a direct hazard for the domain-upsampling recipe of Blakeney et al. [30] that §11 adopts. Hashing the evaluation side into a sorted 64-bit index, rather than the corpus side into a Bloom filter [47], is what makes the rates measurements rather than upper bounds.

Calibration and length normalization. The "calibration, not knowledge" reading of §3 is, stated precisely, a claim about the decision rule rather than about the likelihoods, and the standard tools are those of Guo et al. [48] — temperature scaling and its relatives — as applied to pre-trained transformers by Desai and Durrett [49]. Two features of multiple-choice scoring make the naive version a no-op, and both are known: a single global temperature does not change an argmax, and the quantity being compared across options is a *sum* of log-probabilities whose length normalization is itself the free parameter — the same length bias analysed for translation by Murray and Chiang [51], and the reason Holtzman et al. [50] find that the highest-probability surface form is often not the intended answer. Item 2 of §12 therefore fits the normalization exponent and the length priors *before* the argmax, cross-validated within task, which is the smallest family that contains both the shipped byte-normalized rule and its

plausible competitors. Where we test many tasks at once — the emergence analysis of §9 — we control the false discovery rate by the procedure of Benjamini and Hochberg [52].

Evaluation methodology. The first and most consequential result of this paper concerns the ruler, and here the literature is explicit that the failure modes we diagnose are known ones. Schaeffer et al. [2] argue that the choice between a discrete and a smooth metric can manufacture or dissolve apparent capability jumps—the same split we find between argmax accuracy and the log-likelihood margin (Table 2)—qualifying the emergent-ability claims of Wei et al. [34]. The reproducibility lessons of Biderman et al. [35], from the lm-evaluation-harness, document precisely the normalization and tokenization pitfalls (per-token vs. per-byte length normalization, answer-boundary handling) that our harness correction repairs. That multiple-choice benchmarks carry large seed- and prompt-variance is quantified by Madaan et al. [36] and, for MMLU, by Gema et al. [37]; this is the general form of our BOOLQ coin-flip (Figure 2, Eq. 2). Our use of paired bootstrap confidence intervals follows standard significance-testing practice [38, 39]. Our contribution here is not that benchmarks are noisy but the specific, receipt-backed decomposition (Eq. 1) showing a single unnormalized task dominating our composite, and the rebuilt byte-normalized joint-tokenization scorer that makes a cross-vocabulary comparison legitimate.

14 Conclusion

Training a small model toward the frontier on a fixed rig is governed less by the heroism of one's compute than by three disciplines: measuring the right quantity, sizing the obvious knob against its own noise floor, and acquiring rather than manufacturing one's data. Each result here was produced by adversarial self-audit rather than by further training, and on this rig that method returned more capability-relevant signal per hour than training would have. That is the transferable result, and the paper is the demonstration of it.

Four results stand on their own. *The instrument.* A composite averaging unnormalized per-task margins is dominated by whichever task has the widest natural scale; one benchmark in our suite is a class-prior artifact; and two scoring paths that appear to share a function can silently compute different quantities and normalize by a unit that is not comparable across vocabularies. The rebuilt byte-normalized, joint-tokenization scorer is what makes the cross-vocabulary comparison in Table 2 legitimate at all, and none of the three defects cost a GPU-hour to find. *The learning rate.* An apparent threefold effect resolves, at the level of individual examples, into one benchmark's distribution translating while the other seven stand still — and the obvious alternative explanation, that its two-byte answers put it on a wider scale, is measured and refuted rather than assumed. *Contamination.* The single most benchmark-contaminated corpus in this project, by more than an order of magnitude at matched scanning budget, is not any of its web data but the small buffer hand-assembled for quality and injected exactly where the reference's own recipe says the mixture matters most. Scanned to completion, a component of that reference's published mixture reaches 5.0% on BOOLQ. The mechanism generalises: a quality filter selects for benchmark-shaped text, a mixture inherits the weakest decontamination among its sources, and the annealed end of training is where that lands hardest. *The ledger.* Seventy-six runs yield a measured bound on forecast error — a quarter of the entire improvement the champion's lineage produced — and a receipt-backed data point the emergence debate rarely gets: over a fifteenfold parameter range, tasks are cleared from the smallest model in the record or never cleared at all.

The procedural result is the one we would most like carried elsewhere, because it is free. No number enters a paper without a committed artefact that produces it; no figure is generated from a hardcoded literal; no aggregate is reported over a task list different from the one the sentence beside it describes; and every result *and every number that sizes it* is encoded as a test

that runs on each build. Those rules are enforced by scripts rather than by intention, and the entire paper — every table, every figure, every number — is reproducible on a machine with no GPU at all.

Two lessons about method are worth stating separately. The first is that most of what a laboratory files under *needs a graphics processor* needs instead a scorer that saves its own intermediates. Every scorer in this project persisted the one aggregate it was asked for and discarded the per-choice quantities underneath, so a calibration could not be fitted, a McNemar could not be run and a decontaminated subset could not be re-scored — not because the arithmetic was expensive but because the numbers were gone. A scorer that saves its intermediates costs nothing extra to run and converts four queued GPU experiments into an hour of ordinary processor time. The second is that a queued experiment is worth exactly as much as the last time someone pointed it at real data: a script with a unit-tested core and a criterion fixed in advance is a plan, and the distance between a good plan and a measurement can be three orders of magnitude in cost. Both lessons were cheap to learn and both apply to any laboratory operating under a hard compute ceiling, which is most of them.

References

[1] L. Ben Allal et al. *SmolLM2: When Smol Goes Big.* 2025. arXiv:2502.02737.

[2] R. Schaeffer, B. Miranda, S. Koyejo. *Are Emergent Abilities of Large Language Models a Mirage?* NeurIPS, 2023. arXiv:2304.15004.

[3] N. Muennighoff et al. *Scaling Data-Constrained Language Models.* NeurIPS, 2023. arXiv:2305.16264.

[4] L. Ben Allal et al. *Cosmopedia.* Hugging Face, 2024.

[5] K. Jordan et al. *Muon and modded-nanoGPT.* 2024.

[6] S. Gunasekar et al. *Textbooks Are All You Need.* 2023. arXiv:2306.11644.

[7] Y. Li et al. *Textbooks Are All You Need II: phi-1.5 Technical Report.* 2023. arXiv:2309.05463.

[8] R. Eldan, Y. Li. *TinyStories: How Small Can Language Models Be and Still Speak Coherent English?* 2023. arXiv:2305.07759.

[9] S. Biderman et al. *Pythia: A Suite for Analyzing Large Language Models Across Training and Scaling.* ICML, 2023. arXiv:2304.01373.

[10] Z. Liu et al. *MobileLLM: Optimizing Sub-billion Parameter Language Models for On-Device Use Cases.* ICML, 2024. arXiv:2402.14905.

[11] N. Dey et al. *Cerebras-GPT: Open Compute-Optimal Language Models Trained on the Cerebras Wafer-Scale Cluster.* 2023. arXiv:2304.03208.

[12] J. Kaplan et al. *Scaling Laws for Neural Language Models.* 2020. arXiv:2001.08361.

[13] J. Hoffmann et al. *Training Compute-Optimal Large Language Models.* NeurIPS, 2022. arXiv:2203.15556.

[14] S. Y. Gadre et al. *Language Models Scale Reliably with Over-training and on Downstream Tasks.* 2024. arXiv:2403.08540.

[15] S. Hu et al. *MiniCPM: Unveiling the Potential of Small Language Models with Scalable Training Strategies.* 2024. arXiv:2404.06395.

[16] A. Hägele et al. *Scaling Laws and Compute-Optimal Training Beyond Fixed Training Durations.* NeurIPS, 2024. arXiv:2405.18392.

[17] G. Yang et al. *Tensor Programs V: Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer.* NeurIPS, 2021. arXiv:2203.03466.

[18] J. Liu et al. *Muon is Scalable for LLM Training.* 2025. arXiv:2502.16982.

[19] S. McCandlish, J. Kaplan, D. Amodei, OpenAI Dota Team. *An Empirical Model of Large-Batch Training.* 2018. arXiv:1812.06162.

[20] S. L. Smith, P.-J. Kindermans, C. Ying, Q. V. Le. *Don't Decay the Learning Rate, Increase the Batch Size.* ICLR, 2018. arXiv:1711.00489.

[21] G. Penedo et al. *The FineWeb Datasets: Decanting the Web for the Finest Text Data at Scale.* NeurIPS Datasets & Benchmarks, 2024. arXiv:2406.17557.

[22] J. Li et al. *DataComp-LM: In Search of the Next Generation of Training Sets for Language Models.* NeurIPS Datasets & Benchmarks, 2024. arXiv:2406.11794.

[23] L. Gao et al. *The Pile: An 800GB Dataset of Diverse Text for Language Modeling.* 2020. arXiv:2101.00027.

[24] G. Penedo et al. *The RefinedWeb Dataset for Falcon LLM.* NeurIPS, 2023. arXiv:2306.01116.

[25] L. Soldaini et al. *Dolma: An Open Corpus of Three Trillion Tokens for Language Model Pretraining Research.* ACL, 2024. arXiv:2402.00159.

[26] A. Lozhkov et al. *StarCoder 2 and The Stack v2: The Next Generation.* 2024. arXiv:2402.19173.

[27] K. Lee et al. *Deduplicating Training Data Makes Language Models Better.* ACL, 2022. arXiv:2107.06499.

[28] S. M. Xie et al. *Data Selection for Language Models via Importance Resampling.* NeurIPS, 2023. arXiv:2302.03169.

[29] S. M. Xie et al. *DoReMi: Optimizing Data Mixtures Speeds Up Language Model Pretraining.* NeurIPS, 2023. arXiv:2305.10429.

[30] C. Blakeney et al. *Does Your Data Spark Joy? Performance Gains from Domain Upsampling at the End of Training.* COLM, 2024. arXiv:2406.03476.

[31] G. Hinton, O. Vinyals, J. Dean. *Distilling the Knowledge in a Neural Network.* NeurIPS Deep Learning Workshop, 2015. arXiv:1503.02531.

[32] Y. Kim, A. M. Rush. *Sequence-Level Knowledge Distillation.* EMNLP, 2016. arXiv:1606.07947.

[33] Y. Gu et al. *MiniLLM: On-Policy Distillation of Large Language Models.* ICLR, 2024. arXiv:2306.08543.

[34] J. Wei et al. *Emergent Abilities of Large Language Models.* TMLR, 2022. arXiv:2206.07682.

[35] S. Biderman et al. *Lessons from the Trenches on Reproducible Evaluation of Language Models.* 2024. arXiv:2405.14782.

[36] L. Madaan et al. *Quantifying Variance in Evaluation Benchmarks.* 2024. arXiv:2406.10229.

[37] A. P. Gema et al. *Are We Done with MMLU?* NAACL, 2025. arXiv:2406.04127.

[38] P. Koehn. *Statistical Significance Tests for Machine Translation Evaluation.* EMNLP, 2004.

[39] R. Dror, G. Baumer, S. Shlomov, R. Reichart. *The Hitchhiker's Guide to Testing Statistical Significance in Natural Language Processing.* ACL, 2018.

[40] T. B. Brown et al. *Language Models are Few-Shot Learners.* NeurIPS, 2020. arXiv:2005.14165.

[41] J. Dodge et al. *Documenting Large Webtext Corpora: A Case Study on the Colossal Clean Crawled Corpus.* EMNLP, 2021. arXiv:2104.08758.

[42] O. Sainz et al. *NLP Evaluation in Trouble: On the Need to Measure LLM Data Contamination for each Benchmark.* Findings of EMNLP, 2023. arXiv:2310.18018.

[43] C. Deng et al. *Investigating Data Contamination in Modern Benchmarks for Large Language Models.* NAACL, 2024. arXiv:2311.09783.

[44] S. Golchin, M. Surdeanu. *Time Travel in LLMs: Tracing Data Contamination in Large Language Models.* ICLR, 2024. arXiv:2308.08493.

[45] Y. Oren, N. Meister, N. Chatterji, F. Ladhak, T. B. Hashimoto. *Proving Test Set Contamination in Black Box Language Models.* ICLR, 2024. arXiv:2310.17623.

[46] C. Xu et al. *Benchmark Data Contamination of Large Language Models: A Survey.* 2024. arXiv:2406.04244.

[47] B. H. Bloom. *Space/Time Trade-offs in Hash Coding with Allowable Errors.* Communications of the ACM, 13(7):422–426, 1970.

[48] C. Guo, G. Pleiss, Y. Sun, K. Q. Weinberger. *On Calibration of Modern Neural Networks.* ICML, 2017. arXiv:1706.04599.

[49] S. Desai, G. Durrett. *Calibration of Pre-trained Transformers.* EMNLP, 2020. arXiv:2003.07892.

[50] A. Holtzman, P. West, V. Shwartz, Y. Choi, L. Zettlemoyer. *Surface Form Competition: Why the Highest Probability Answer Isn't Always Right.* EMNLP, 2021. arXiv:2104.08315.

[51] K. Murray, D. Chiang. *Correcting Length Bias in Neural Machine Translation.* WMT, 2018. arXiv:1808.10006.

[52] Y. Benjamini, Y. Hochberg. *Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing.* JRSS-B, 57(1):289–300, 1995.