DOES RANDOM SEARCH DISCOVER TASK EXPERTS? REVISITING NEURAL THICKETS

Anonymous authors
Paper under double-blind review

ABSTRACT

Pretrained models are typically viewed as generalists, requiring bespoke post-training to achieve task-specific expertise. Gan & Isola (2026) recently proposed a striking alternative: that pretrained models lie in a dense "neural thicket" of task specialists, which can be surfaced with simple Gaussian weight perturbations. We examine this neural thicket of task-specific experts and identify several key findings which question this perspective. First, we find that the apparent lift of the best-of-N selected perturbation decays quickly with the size m of the selection set, strongly suggesting overfitting to the selection questions. In fact, for 10 of the 12 model–task pairs we study, a simple null model of pure selection noise predicts more than 75% of the top expert's reward lift. Second, the top experts' advantage often vanishes on held-out questions, or remains within the run-to-run variability of the unperturbed pretrained model. Third, we find that majority voting over the top-K selected perturbations—coined RandOpt(N, K) by Gan & Isola (2026)—is often matched or outperformed by majority voting over K samples from the base model. Overall, our results suggest that random weight perturbations may be ineffective at uncovering task experts around pretrained models, and that adapting pre-trained models to downstream tasks requires more sophistication than random search.

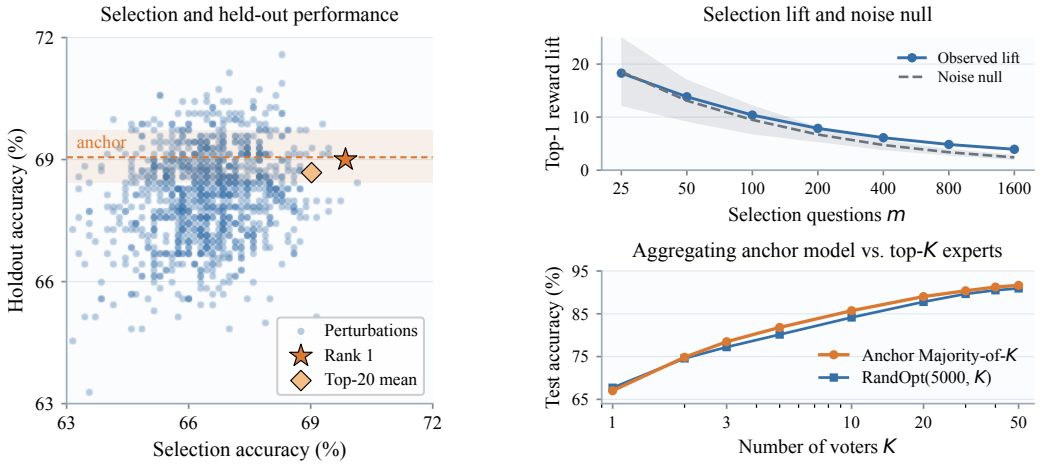

Figure 1: OLMo-3-7B-Instruct on Countdown. **Left:** Performance of $N = 1,000$ perturbations at $\sigma = 0.001$; the anchor line/band displays its five-run mean/range on a fixed runtime. Note little relation between selection and held-out performance. **Top right:** Reward difference between the best-performing perturbation ("top-1 expert") at $N = 1,000$ and the perturbation population mean on the selection set of size m. A null model of pure evaluation noise (shown with ± 2 SD) predicts most observed lift, e.g., 85% at $m = 200$. **Bottom right:** RandOpt versus anchor majority-of-K, at $N = 5,000$ and $\sigma \in \{0.5, 1, 2\} \times 10^{-3}$. Anchor voting leads at every $K \geq 3$.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

1 INTRODUCTION

Post-training stands as a sprawling and often expensive component of language-model development. Techniques such as instruction tuning (Wei et al., 2022), preference optimization (Rafailov et al., 2023), and reinforcement learning with learned rewards (Ouyang et al., 2022) all serve to improve a pretrained model's performance on a particular task, often with substantial demands on data and compute. Against this backdrop, Gan & Isola (2026) proposed a surprising shortcut based on random search. Their method, RandOpt(N, K), begins with a selection set S_{sel} of questions for a particular task $\mathcal{T}$. It then evaluates N Gaussian weight perturbations $\{\theta_i\}_{i \leq N}$ of the pretrained model θ on S_{sel}, and preserves the top-K performing perturbations. In particular, each perturbation is generated by adding i.i.d. Gaussian noise to each parameter of the pretrained model: $\theta_i = \theta + \mathcal{N}(0, \sigma^2)$. At test time, RandOpt$(N, K)$ aggregates the K selected models by majority vote. Each perturbed model θ_i is then thought of as an attempt at exhibiting a task expert for $\mathcal{T}$; the top such experts ought to compete with a genuinely post-trained version of θ.

Surprisingly, Gan & Isola (2026) reported that this simple form of random search can produce task-specific experts. On the Countdown arithmetic task, for instance, they report that a single top-selected perturbation of OLMo-3-7B-Instruct achieves about 74% accuracy, compared with about 65% for the unperturbed "anchor" model. At the ensemble level, they report that RandOpt can even outperform reinforcement-learning baselines on reasoning tasks such as GSM8K and MBPP (Gan & Isola, 2026, Table 4). The authors attribute this performance to a *neural thicket*: a dense population of task-specific experts surrounding the pretrained weights. In their account, pretraining balances multiple task objectives, and a model optimized for the aggregate loss may remain close to better solutions for individual tasks. The neural-thicket hypothesis further proposes that such experts are sufficiently abundant for random perturbations alone to recover them, without additional training.

However, several important aspects of the neural thicket merit further study. First, Gan & Isola (2026) primarily report the performance of individual perturbations (i.e., top experts) on the selection set, often reserving the held-out test set for the final RandOpt ensemble. This leaves it somewhat unclear the extent to which the top-expert advantage arises from selection effects, rather than a persistent performance advantage. A priori, these effects could be substantial for the setup of Gan & Isola (2026), which selects among thousands of perturbations using only $m = 200$ selection questions. Furthermore, it remains somewhat unclear the extent to which RandOpt's performance is driven by the advantage (and/or diversity) of the top-K experts, as opposed to the simple benefits of majority voting over K samples from the anchor model. This motivates us to closely study two questions: **Q1**: How much of the apparent advantage of top-ranking experts persists on held-out questions, as opposed to arising from selection effects? **Q2**: How much does majority voting over the top-K experts improve performance over majority voting over K samples from the anchor model?

We begin our study of Q1 by testing a null hypothesis of pure selection noise. In this model, each perturbed model has the same expected performance as the anchor, but suffers *independent* noise when evaluated at each question. (Intuitively, the anchor model is a random variable, and each perturbed model is one crystallization of that random variable.)[1] A crucial feature of this model—in which there is no neural thicket of task experts—is that the top expert's performance on the selection set S_{sel} should decay quickly as a function of its size, $m = |S_{\text{sel}}|$. Intuitively, a larger selection set reduces the variance of each expert's performance, reducing the chances of seeing a very lucky expert. In contrast, were there to exist an expert with a persistent performance advantage, its expected lift over the perturbation population would remain stable as m grows.

We find that the null model indeed predicts a large majority of the top experts' apparent advantage on the selection set, and that this advantage decays quickly in m (Fig. 1, top right). For instance, at $m = 200$, among 1,000 OLMo-3-7B-Instruct perturbations on Countdown, the null predicts 6.72 of the 7.89 reward points gained over the perturbation average—approximately 85%. Furthermore, the decline in the top expert's apparent advantage (on the selection set) continues well beyond the $m = 200$ questions used in Gan & Isola (2026). On Countdown, OLMo sees its top expert advantage drop from 7.89 points at $m = 200$ to 3.94 at $m = 1600$. This strongly suggests that at least half of the top expert's apparent advantage is due to selection effects.

[1] One could also think of the anchor model as being stochastic, and of each perturbed model as a single rollout. Our primary experiments use greedy decoding, however, in keeping with Gan & Isola (2026).

We then examine performance on held-out questions. For OLMo-3-7B-Instruct across the Countdown, GSM8K, and MBPP reasoning tasks, we find that the rank-1 experts' advantage nearly vanishes on held-out data, with a mean gain of only 0.16 accuracy points across the three tasks. Notably, this lands within the run-to-run variability of the anchor model on the same held-out questions (even with greedy decoding!), suggesting that top experts differ little from the anchor. On Countdown, the 0.44-point gain over the anchor from our full $N = 5,000$ candidate search falls far short of the single-expert improvement reported by Gan & Isola. We discuss this reproduction gap and our runtime configuration in Appendices B and D.2.

We also examine the Qwen2.5-Instruct family of models at the 3B, 7B, and 32B scales. We find that their experts struggle similarly on GSM8K and MBPP, with held-out gains of only about 1%. On Countdown, however, Qwen-3B and 7B retain considerable gains of 12.16 and 11.41 accuracy points, respectively, over their anchors. Notably, about 83–94% of these gains are driven by improved validity of generated expressions, rather than improved arithmetic ability. That is, the top experts produce valid expressions more often than the anchor, which increases accuracy even if the conditional correctness of valid expressions remains similar. Overall, for 10 of the 12 model–task settings we consider, our results strongly suggest that the apparent advantage of top experts is largely due to selection effects, and that held-out performance is often within the run-to-run variability of the anchor model.

We then turn to Q2, and compare RandOpt's majority-vote aggregation of the top-K experts with simple test-time majority voting (TT-MV) over K samples from the anchor model. At $K = 50$, across the OLMo-3-7B-Instruct and Qwen2.5-$\{$3B,7B,32B$\}$-Instruct models and the tasks $\{$Countdown, GSM8K, MBPP$\}$, we find that RandOpt is statistically indistinguishable from TT-MV in 10 of the 12 model–task settings. In the remaining 2 settings—Qwen2.5-Instruct at 7B and 32B on Countdown—RandOpt slightly outperforms TT-MV by about 2% (Table 5). Our results suggest that in most cases, the benefits of RandOpt's majority-vote aggregation are largely explained by the benefits of majority voting over the anchor model. Overall, our results question the existence of a dense neural thicket, and suggest that eliciting genuine task expertise from pretrained models requires more sophistication than random search.

2 Preliminaries and Methodology

2.1 The neural thicket

Let us briefly review the neural thicket findings reported by Gan & Isola (2026). We use their language throughout the paper, referring to pretrained models as "anchors," and to top-performing weight perturbations of pretrained models as "experts."

Claim 1: Task experts are dense in weight space. The first and most central claim of Gan & Isola (2026) is that pretrained models are closely surrounded in weight space by task experts. This effect is sufficiently pronounced that simple Gaussian weight perturbations of the pretrained model can surface task experts with shockingly high probability (e.g., $> 10\%$). In fact, on some model–task pairs, as many as 45% of random weight perturbations produce a task expert that outperforms the anchor by 15+%! (See their Figure 3(a).) This suggests, strikingly, that pretraining produces not just the weight vector of a single performant model, but an entire weight-space neighborhood hosting a "veritable thicket of task-specific solutions" (Gan & Isola, 2026, p. 1).

Claim 2: RandOpt competes with post-training. Perhaps the second major claim of Gan & Isola is that simple aggregations of such experts can be used to match or even outperform standard post-training methods. For instance, for OLMo-3-7B-Instruct on the Countdown reasoning task, it is reported that an ensemble of the top 50 experts—i.e., RandOpt with $K = 50$—outperforms ensembles of GRPO post-trained models by 5 points (their Figure 1(c)).

Claim 3: Expert density increases with model size. Finally, Gan & Isola report that the density of task experts increases with model size. Roughly speaking, the neural thicket can sometimes be difficult to observe for models at the 0.5B or 1.5B scales, but appears fairly consistently at the 3B+ and 7B+ scales. In fact, when measuring the proportion of random perturbations that outperform the anchor, it can appear to increase consistently with model size (see their Figures 3(a) and 8). This points to an intriguing scaling axis of the neural thicket effect that merits further study.

2.2 RandOpt and the evaluation setting

We study three reasoning tasks used by Gan & Isola: Countdown, GSM8K, and MBPP. Countdown asks for an expression reaching a target integer using each of 3–4 supplied integers exactly once, and operations from $\{+, -, \times, \div\}$. GSM8K (Cobbe et al., 2021) comprises grade-school word problems requiring two to eight arithmetic steps and a numerical answer. MBPP (Austin et al., 2021) asks for Python programs satisfying natural-language specifications and three tests.

The pretrained models in our main searches are OLMo-3-7B-Instruct and Qwen2.5-Instruct at the 3B, 7B, and 32B sizes (Team OLMo, 2025; Qwen Team, 2025). Recall that RandOpt(N, K) samples N candidates around an anchor with weights θ,

$$\theta_i = \theta + \sigma_i \, \varepsilon_i, \qquad i \in [N], \tag{1}$$

where σ_i controls the perturbation size and ε_i is standard Gaussian noise drawn independently for each entry in θ. RandOpt(N, K) then ranks all candidates by their mean reward on m selection questions and retains the top K. At test time, it aggregates these top-K models via majority vote over valid extracted answers. We use the released RandOpt implementation, which samples each $\sigma_i \in \{0.5, 1, 2\} \times 10^{-3}$ uniformly at random. For selection, Countdown uses answer correctness plus 0.1 times a format score, while GSM8K and MBPP use binary task rewards; see Appendix B.

For OLMo and Qwen-3B/7B, we evaluate $N = 5,000$ candidates on $m = 200$ selection questions; for Qwen-32B, we use $N = 1,000$. We then evaluate the top 50 on 1,400 held-out Countdown questions, 1,319 GSM8K questions, and 500 MBPP tasks. Following Gan & Isola (2026), we default to greedy decoding throughout.

2.3 Measuring selection and transfer

We first examine a statistical model of expert performance under the null hypothesis that candidates have no persistent differences in expected reward. Selecting among $N = 5{,}000$ candidates can then yield a substantial apparent advantage through finite-sample variation alone. In particular, under the *selection-noise null*, the reward Y_{ij} of perturbation i on question j follows

$$Y_{ij} = \mu_j + \varepsilon_{ij}, \qquad \mathbb{E}[\varepsilon_{ij}] = 0, \quad \mathrm{Var}(\varepsilon_{ij}) = v_j. \tag{2}$$

Here μ_j is the common expected reward on question j and the errors ε_{ij} are independent across candidate–question pairs. Letting $S_i(m)$ denote candidate i's mean reward on m selection questions, we define the top expert's performance lift above the perturbation-population mean as

$$\Delta_{\mathrm{sel}}(m) = \max_i S_i(m) - \frac{1}{N} \sum_{i=1}^{N} S_i(m). \tag{3}$$

We implement the empirical null by sampling each Y_{ij} with replacement from the observed candidate rewards on question j. Averaging over a fixed set of m selection questions gives $\mathrm{Var}(S_i(m)) = v/m$, where v is the mean of the corresponding per-question variances v_j. Approximating the candidate means as Gaussian[2] then gives

$$\mathbb{E}[\Delta_{\mathrm{sel}}(m)] \approx a_N \sqrt{v/m}, \qquad a_N = \mathbb{E}\left[\max_{i \le N} Z_i\right], \quad Z_i \sim \mathcal{N}(0, 1). \tag{4}$$

Here a_N is the expected maximum of N independent standard normal variables (i.e., its Nth order statistic); for example, $a_{1000} \approx 3.241$ and $a_{5000} \approx 3.678$. A key feature of the null model is that for fixed N, the expected lift $\mathbb{E}[\Delta_{\mathrm{sel}}(m)]$ vanishes as $m \to \infty$. (Intuitively, every perturbed model's mean reward simply converges to the shared population mean.) In contrast, if there exists any candidate with a persistent reward advantage of $\varepsilon > 0$, then the limiting expected lift is at least ε.

3 Selection, Transfer, and Test-Time Voting

3.1 Understanding selection lift and held-out transfer

Top-1 expert lift is largely predicted by selection noise. We begin by comparing the selection-noise null (Section 2.3) with the observed reward lift of the top-selected, or "rank-1," expert. We find

[2]Assuming independent, bounded question-level rewards, the central limit theorem implies that candidate means are asymptotically Gaussian as $m \to \infty$. See Appendix C for further detail on the null model.

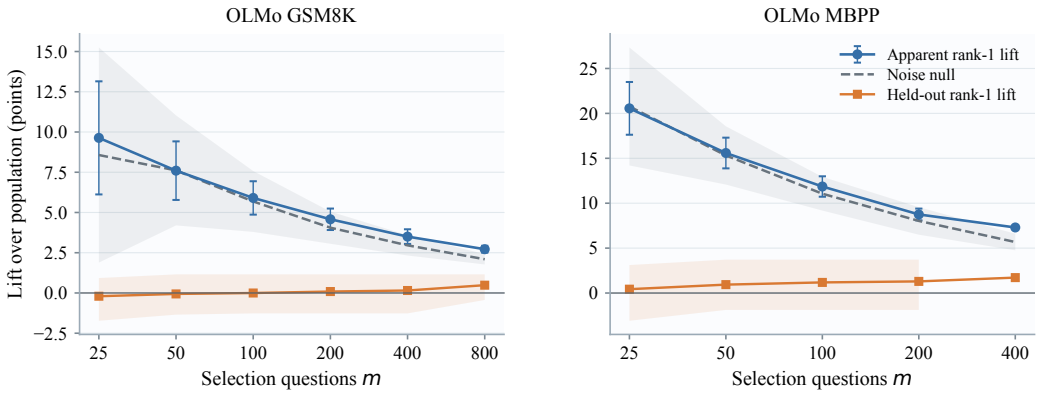

Figure 2: Selection-size sweeps for OLMo-3-7B-Instruct, $N = 5000$. Blue: rank-1 selection reward lift over the perturbation population, with ± 1 SD across selection subsets. Gray: empirical noise null with ± 2 simulation SD. Orange: held-out rank-1 accuracy gain over the perturbation population. **Left:** GSM8K shows little transfer. **Right:** MBPP has a positive effect at larger m on the official 500-task test. ($m = 400$ uses the entire selection pool, thus lacks selection-subset bands.)

that on each of Countdown, GSM8K, and MBPP, more than 85% of OLMo's observed rank-1 lift is predicted by the null model (Figures 1 (top right) and 2). This holds for $m = 200$, as used by Gan & Isola (2026). For Qwen2.5 at 3B, 7B, and 32B, the null predicts approximately 76–86% of the rank-1 lift on GSM8K and 83–100% on MBPP. Countdown, however, retains larger residuals at 3B and 7B: the null predicts only 31% and 53% of their observed lifts, respectively (Appendix Table 12).

OLMo expert lift decays with selection-set size. We next examine how the rank-1 expert's apparent advantage varies as a function of the selection-set size m. A persistent expert advantage would be expected to remain stable as m increases, whereas an advantage decaying in m would strongly suggest (some) overfitting of the top-1 expert to the selection questions. We find that top expert lift indeed decreases sharply with m. For OLMo on Countdown, increasing m from 200 to 1600 halves the rank-1 selection lift, from 7.89 to 3.94 reward points; GSM8K and MBPP see similar drops (Figures 1, 2). Furthermore, OLMo's rank-1 experts gain little held-out accuracy over the perturbation population on GSM8K and MBPP (Figure 2).

Small Qwen models display a persistent advantage. In contrast to OLMo, we find that Qwen-3B and 7B retain their top experts' advantage on Countdown as m increases. Respectively, their rank-1 lifts decrease only slightly from 24.1 and 18.2 points at $m = 200$ to 23.8 and 16.1 at $m = 800$. This suggests the existence of a persistent advantage enjoyed by their top experts. We investigate the source of this advantage further at the end of Section 3.2. On GSM8K and MBPP, however, Qwen-3B, 7B, and 32B all see their top experts' advantage decay with m, suggesting that the apparent advantage includes selection effects (Appendix Table 11).

3.2 Comparing experts with the anchor

OLMo's top experts see mixed performance. We now compare selected experts with their anchor models on held-out data (Figure 3). For OLMo, we see its rank-1 experts change test accuracy by $+0.44$ points on Countdown, -2.03 on GSM8K, and $+2.08$ on MBPP (Figure 3). Averaging the accuracies across the top 20 experts, we find that OLMo's top experts achieve average gains over the anchor of $+0.73$ points on Countdown, -0.44 on GSM8K, and $+1.53$ on MBPP. In short OLMo's top experts see generally mixed performance across tasks; in Appendix D.1 we study the sensitivity of these findings to the choice of selection subset, and find them to be fairly stable.

Most Qwen experts differ little from the anchor. On the GSM8K held-out set, the rank-1 experts from Qwen-3B, 7B, and 32B change accuracy by a relatively modest $+0.58$, $+0.38$, and $+1.93$ points, respectively. On MBPP, meanwhile, we find the rank-1 expert performance to be mixed, with anchor differences of $+0.52$, -1.88, and -2.76 points, respectively. Averaging over the top-20 experts, we find that Qwen-3B, 7B, and 32B change accuracy by $+0.24$, -0.11, and $+1.95$ points on

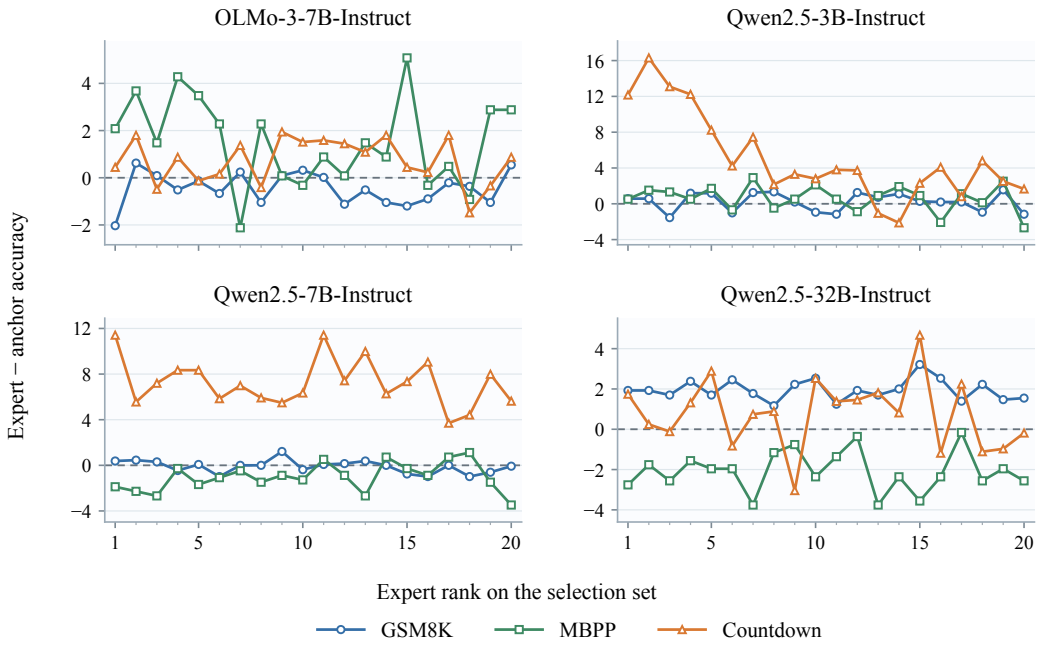

Figure 3: Held-out performance of the top-20 selected perturbations, expressed as percentage point accuracy difference from the anchor. Anchor baselines are averaged over five runs per model–task. All searches use $N = 5,000$ candidates, with the exception of Qwen 32B, which uses $N = 1,000$. GSM8K and MBPP generally see little advantage from top experts, while Countdown sees substantial gains for Qwen-3B and 7B, but not for OLMo or Qwen-32B.

GSM8K, and by $+0.62$, -1.03, and -2.08 points on MBPP, respectively. For these tasks, the Qwen models do not appear to benefit appreciably from random weight search (at $N = 5,000$).

Qwen 3B and 7B see substantial gains on Countdown, with a catch. On Countdown, we find that smaller Qwen models can surface experts with considerable held-out performance gain. In particular, Qwen-3B's rank-1 expert reaches 26.8% accuracy on the held-out data against its anchor's 14.6%—a striking 12.2-point gain—and its top-20 experts average a 5.1 point gain. Similarly, Qwen 7B's rank-1 expert improves upon the anchor by an impressive 11.4 accuracy points.

However, we find that *a substantial part of Countdown gains comes from answering more questions, not solving them better*. Countdown credits a question only when a valid expression is extracted, so an expert can gain either by solving attempted questions more accurately or simply by attempting more of them. Splitting held-out questions by which model produces a valid expression makes the distinction exact. For Qwen-3B, whose anchor emits a valid expression on only 26% of questions, the rank-1 expert's entire +11.4-point gain comes from attempting more: it answers 345 questions the anchor leaves blank (converting 67.0%, near the anchor's own per-attempt rate) and abandons 181 the anchor had answered, while on the 185 questions both models answer it wins 28 and loses 26, essentially 50-50 (McNemar $p = 0.89$). Qwen-7B repeats the pattern with one difference: roughly 86% of its net gain likewise comes from attempting more questions, but on the 482 questions both models answer it wins 58 and loses 38 ($p = 0.052$), weak evidence of a small genuine "reasoning-relevant" improvement. See Appendix D.3.1 for further detail.

3.2.1 USPTO: EXPLOITING LABEL IMBALANCE

We also examine OLMo's neural thicket for the USPTO-50k chemistry classification task, where one must assign a chemical reaction to one of ten numbered categories (Schneider et al., 2016). Crucially, the categories are imbalanced; category 1 covers 30% of questions. In fact, this surpasses OLMo's anchor performance of 27.5% on the held-out set, meaning that the "predict class 1" strategy outperforms the anchor!

		$K = 10$		$K = 20$		$K = 50$	
Model	Task	Difference	Significant	Difference	Significant	Difference	Significant
OLMo 7B	GSM8K	$+0.41$	No	$+0.30$	No	$+0.56$	No
	MBPP	$+0.87$	No	$+0.72$	No	$+0.53$	No
	Countdown	-1.57	Yes	-1.24	Yes	-0.71	No
Qwen 3B	GSM8K	-0.71	No	0.00	No	-0.61	No
	MBPP	$+0.66$	No	-0.03	No	$+0.87$	No
	Countdown	$+7.01$	Yes	$+2.57$	Yes	-0.84	No
Qwen 7B	GSM8K	-0.84	Yes	-0.36	No	-0.28	No
	MBPP	-2.12	No	-0.87	No	-1.94	No
	Countdown	$+2.17$	Yes	$+2.73$	Yes	$+2.25$	Yes
Qwen 32B	GSM8K	$+0.03$	No	$+0.24$	No	$+0.32$	No
	MBPP	-0.84	No	$+0.85$	No	$+0.09$	No
	Countdown	$+1.50$	Yes	$+2.27$	Yes	$+1.86$	Yes

Table 5: Held-out accuracy of RandOpt minus that of anchor TT-MV, in percentage points at the voting budgets $K = \{10, 20, 50\}$. Statistical significance columns record whether a two-sided 95% paired-question bootstrap interval excludes zero. On GSM8K and MBPP, TT-MV matches or outperforms RandOpt. On Countdown, Qwen models prefer RandOpt and OLMo prefers TT-MV.

At first glance, OLMo's top experts appear to achieve considerable gains over the anchor, with an average of 5.06 points across the top 20 experts (Table 4). However, upon randomly reassigning category numbers per question, we find that about 80% of the apparent gains disappear. Notably, the anchor's accuracy remains nearly constant under this shuffling, while the top expert's absolute accuracy drops considerably. This suggests that the apparent thicket is largely due to the shortcut of exploiting label imbalance, rather than "genuine" task expertise. Nevertheless, a modest hold-out advantage of 1-3% persists, though it remains below the accuracy of the "predict class 1" strategy.

Answer choices	Top-1	Top-5	Top-10	Top-20
Original	$+7.79$	$+5.21$	$+5.42$	$+5.06$
Shuffled	$+3.20$	$+1.48$	$+1.22$	$+1.12$

Table 4: OLMo-3-7B-Instruct at $N = 5,000$ perturbations on the USPTO-50k dataset. The first row reports top-experts' held-out gain over the anchor in accuracy points. The second reports the same for held-out data that has its category numbers shuffled. Roughly 60–80% of top expert performance vanishes under shuffling, suggesting that experts primarily learn to abuse label imbalance.

3.3 RandOpt versus anchor test-time voting

OLMo's TT-MV exceeds RandOpt on Countdown. In Gan & Isola (2026)'s headline setting of OLMo on Countdown, we find that anchor majority voting outperforms RandOpt for every evaluated $K \geq 3$ (Figure 1, bottom right). At $K = 1$ (i.e., selecting the top expert), RandOpt exceeds the anchor by just 0.44 points, which lands within the roughly 1.5-point range of variability for repeated anchor evaluations. Notably, this falls far short of the approximately 8-point single-expert gain reported by Gan & Isola (2026), which we were not able to reproduce using their released code.

Qwen sees RandOpt gains on Countdown. For Qwen-7B and 32B on Countdown, RandOpt exceeds anchor voting at all three budgets $K \in \{10, 20, 50\}$, with fairly stable gains of 1.5–2.7 accuracy points that are significant at the 5% level (Table 5). In contrast, Qwen 3B exhibits a strong advantage for RandOpt of about 7 points at $K = 10$, which narrows to 2.6 points at $K = 20$ before falling to -0.8 points at $K = 50$. This pattern is consistent with Qwen-3B's strong leading experts and rapidly diminishing gains at lower ranks (Fig. 3).

TT-MV matches RandOpt across GSM8K and MBPP. Across GSM8K and MBPP, we find that RandOpt never significantly exceeds anchor TT-MV at $K \in \{10, 20, 50\}$ (Table 5); gaps range from

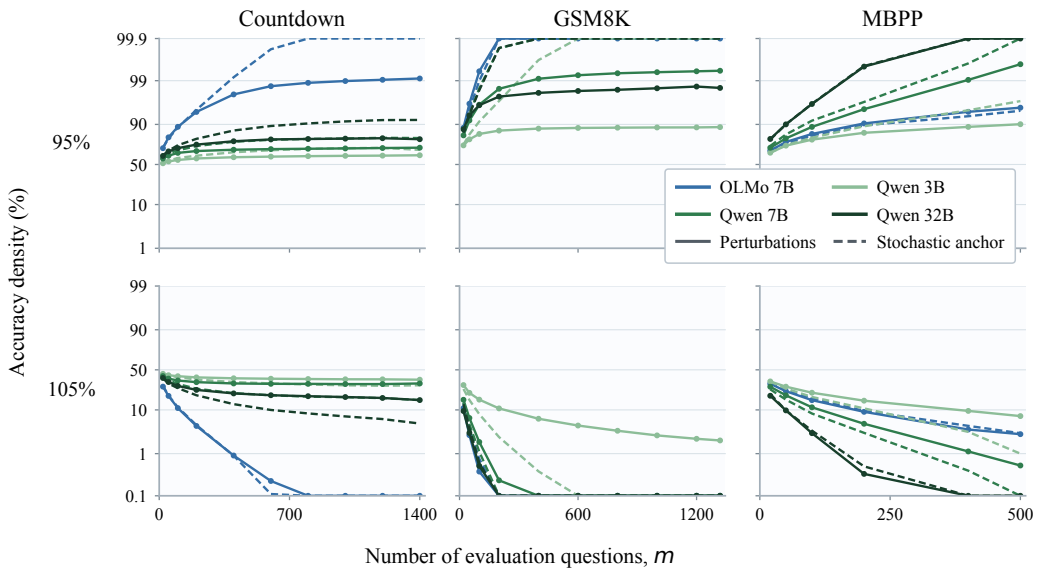

Figure 6: Accuracy density versus evaluation-set size for OLMo and Qwen models. Each row gives the fraction of accuracies exceeding the indicated percentage of their population mean on the same questions. (I.e., 95% for the top row and 105% for the bottom row.) Accuracy values are clipped to $[0.1\%, 99.9\%]$ for display. Solid curves use perturbations at $\sigma = 10^{-3}$ with greedy decoding; dashed curves use 100 rollouts from the anchor model with temperature 0.6 and top-$p = 0.95$. Accuracy density often trends toward 0 or 1 with increasing m, suggesting that perturbations concentrate around their population mean, rather than exhibiting a persistent advantage. Stochastic rollouts from the anchor model display qualitatively similar accuracy trends, suggesting that the observed behavior may be due to mere sampling noise.

-2.12 to $+0.87$ accuracy points. In fact, the only statistically significant difference favors anchor voting on Qwen-7B GSM8K at $K = 10$, by 0.84 points. Anchor voting thus provides a strong no-search baseline, matching or exceeding RandOpt on most model–task pairs.

3.4 Increasing diversity of the neural thicket

We next examine Gan & Isola's claim that the density of task experts increases with model size. For a given evaluation metric (e.g., accuracy) and threshold value (e.g., 105%), the *density* of a model's neural thicket is the fraction of random perturbations that exceed the threshold relative to the population mean of perturbations.[3]

Diversity decays with evaluation-set size. We first observe that for a fixed choice of threshold $\tau > 100\%$, the density of perturbations exceeding τ typically decays to 0 with the number of evaluation questions m (Figure 6). Similarly, for $\tau < 100\%$, the density often tends to 1 with increasing m. This behavior appears largely consistent across models on MBPP and GSM8K, and for OLMo on Countdown. However, the Qwen models would appear to display persistent behavior on Countdown, with a stable proportion of favorable and unfavorable perturbations as m increases. This is consistent with our previous finding that experts from Qwen-3B and 7B can exhibit a persistent advantage on Countdown, while OLMo and the larger Qwen models do not. Notably, however, we also observe that *qualitatively similar trends appear even for stochastic rollouts from the anchor model*. This demonstrates that the observed trends may be an artifact of mere sampling noise, rather than persistent advantage among a subpopulation.

Diversity does not appear to increase with model size. We next examine the density of perturbations exceeding a given threshold τ as a function of model size (Figure 7). For thresholds $\tau > 100\%$,

[3]We note that Gan & Isola define density relative to the anchor model, rather than the perturbation population. We find that using the perturbation population mean helps to isolate the behavior of the perturbations themselves.

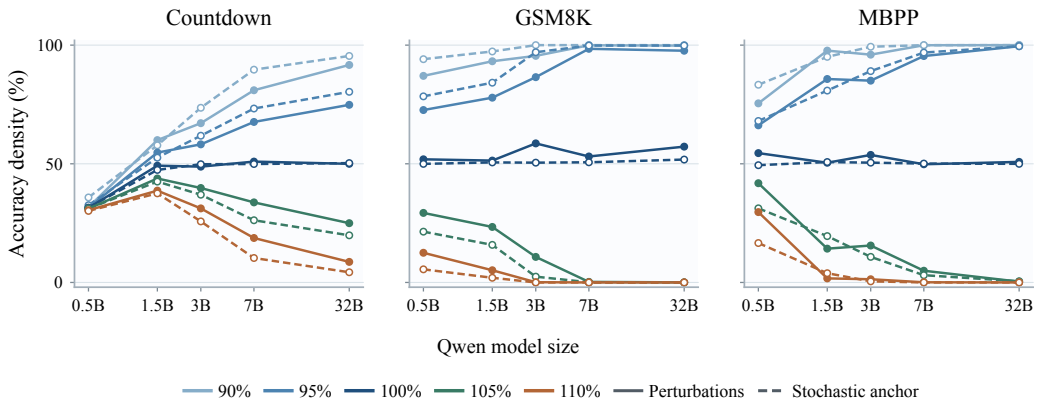

Figure 7: Accuracy density versus Qwen model size on $m = 200$ questions. Accuracy density trends toward 0 (respectively 1) for thresholds $\tau > 100\%$ (respectively $\tau < 100\%$), suggesting simple concentration of perturbations around their population mean. Similar trends are observed for simple stochastic rollouts from the anchor model.

we find that the density of favorable perturbations typically decreases with model size, contrary to Gan & Isola's claim. Furthermore, the behavior of stochastic rollouts from the anchor model is again qualitatively similar to that of the perturbations, suggesting that mere sampling noise may be responsible for the observed trends.

4 Technical Details

Let us briefly remark upon two subtleties encountered in our study of the neural thicket.

Greedy decoding is nondeterministic. We observe that even with greedy decoding, inference of the anchor model is not deterministic. In the case of OLMo-3-7B-Instruct on Countdown, across five executions of the anchor on a single runtime, we find the mean final-answer disagreement proportion to be 6.1%, with an accuracy range of about 1 point across the runs. Thus, in practice, even a genuine (mild) persistent advantage exhibited by a top expert may be obscured by the run-to-run variability of the anchor model. We provide further detail on variability of anchor models in Appendix D.2.

Tied weight perturbations and restoration artifacts. The released neural thicket implementation contains two artifacts that, a priori, may affect the behavior of top experts. First, the generator for perturbing each weight tensor has its random seed reset before perturbing each tensor, causing equal-shaped tensors to receive identical noise. Second, the anchor restoration step subtracts the perturbation in bf16 arithmetic, which can leave a rounding residual that accumulates across candidates (i.e., random walk behavior). We ablate each of these artifacts in Appendix E, and find them to have little effect on our reported results.

5 Discussion and Conclusion

We study the neural thicket recently proposed by Gan & Isola (2026), which posits that pretrained models lie in a region of weight space densely populated by task experts. In most settings we consider, we find that the apparent advantage of top-ranking experts on the selection set is largely explained by selection effects, and decays quickly in the size of the selection set. On held-out questions, we find that top-expert performance often differs little from that of the unperturbed anchor model.

Overall, our results suggest that the weight space neighborhood of pretrained models can often fail to exhibit a neural thicket. Relatedly, they suggest that elicitation of task expertise in a pretrained model can be a non-trivial endeavor, requiring more than simple random search. Understanding the conditions under which a neural thicket may exist—and for which task-expertise follows easily from pretraining—remains an interesting question.

AI USE STATEMENT

AI tools, including OpenAI's Codex and Anthropic's Claude Code, were used to assist with experimentation and analysis code, figure generation, literature review, and manuscript editing. Portions of the appendix were also drafted with AI assistance. AI assisted in analyzing the selection noise null model, though it was suggested by the authors. The authors assume responsibility for all content and claims in the paper.

REPRODUCIBILITY STATEMENT

Precise experimental details are provided in Appendix B, including model perturbation code—taken from the original neural thicket implementation—and data / inference details. The exact experiment and analysis code, along with the evaluations reported in the paper, are currently being prepared for release.

REFERENCES

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 7319–7328. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.acl-long.568. URL https://aclanthology.org/2021.acl-long.568/.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc V. Le, and Charles Sutton. Program synthesis with large language models, 2021. URL https://arxiv.org/abs/2108.07732.

Yunpeng Ba, Zhi Zheng, Yue Xie, Jiaqing Li, Xialiang Tong, Tao Zhong, Mingxuan Yuan, Zhichao Lu, Xuyang Wu, and Zhenkun Wang. Understanding evolution strategies for LLM reasoning: Broader reasoning coverage than GRPO, 2026. URL https://arxiv.org/abs/2608.27351.

Bradley Brown, Jordan Juravsky, Ryan Ehrlich, Ronald Clark, Quoc V. Le, Christopher Ré, and Azalia Mirhoseini. Large language monkeys: Scaling inference compute with repeated sampling. *arXiv preprint arXiv:2407.21787*, 2024. URL https://arxiv.org/abs/2407.21787.

Nigel Bastian Cendra, Abdelhamid Ezzerg, Fernando Julio Cendra, Jeremias Knoblauch, and Jakob Zeitler. Can Bayesian Optimization efficiently find a strong single expert in Neural Thickets?, 2026. URL https://arxiv.org/abs/2608.10867.

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

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1383–1392. Association for Computational Linguistics, 2018. doi: 10.18653/v1/P18-1128. URL https://aclanthology.org/P18-1128/.

Yulu Gan and Phillip Isola. Neural thickets: Diverse task experts are dense around pretrained weights, 2026. URL https://arxiv.org/abs/2603.12228.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626, 2023. URL https://arxiv.org/abs/2309.06180.

Jack Lu, Zhenbang Yang, Mike Lasby, Tejas Pote, Yani Ioannou, and Mengye Ren. The surprising effectiveness of deleting weights in LLM reasoning and adaptation. In *ICML Workshop on Foundations of Deep Generative Models: Understanding Memorization, Generalization, and Reasoning*, 2026. URL https://openreview.net/forum?id=JnfolztAV3.

Miroslav Lžičař. Auditing Neural Thickets with low-rank routes. In *ICML Workshop on Weight-Space Symmetries: From Foundations to Practical Applications*, 2026. URL https://openreview.net/forum?id=Xe8pKZ1Xdk.

Alireza Mehrtash, Purang Abolmaesumi, Polina Golland, Tina Kapur, Demian Wassermann, and William M. Wells, III. PEP: Parameter ensembling by perturbation. In *Advances in Neural Information Processing Systems*, volume 33, pp. 8895–8906, 2020. URL https://papers.nips.cc/paper/2020/hash/652c208b21f13f6e995bfc1154a1a2e5-Abstract.html.

John X. Morris, Niloofar Mireshghallah, Mark Ibrahim, and Saeed Mahloujifar. Learning to reason in 13 parameters, 2026. URL https://arxiv.org/abs/2602.04118.

Declan Oller, Tobias Glasmachers, and Giuseppe Cuccu. Analyzing reinforcement learning benchmarks with random weight guessing. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 975–982. International Foundation for Autonomous Agents and Multiagent Systems, 2020. URL https://exascale.info/assets/pdf/oller2020aamas.pdf.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F. Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/b1efde53be364a73914f58805a001731-Abstract.html.

Xin Qiu, Yulu Gan, Conor F. Hayes, Qiyao Liang, Yinggan Xu, Roberto Dailey, Elliot Meyerson, Babak Hodjat, and Risto Miikkulainen. Evolution strategies at scale: LLM fine-tuning beyond reinforcement learning. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2509.24372.

Qwen Team. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D. Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, pp. 53728–53741, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a85b405ed65c6477a4fe8302b5e06ce7-Abstract-Conference.html.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017. URL https://arxiv.org/abs/1703.03864.

Nadine Schneider, Nikolaus Stiefl, and Gregory A. Landrum. What's what: The (nearly) definitive guide to reaction role assignment. *Journal of Chemical Information and Modeling*, 56(12): 2336–2346, 2016. doi: 10.1021/acs.jcim.6b00564. URL https://pubs.acs.org/doi/10.1021/acs.jcim.6b00564.

Team OLMo. OLMo 3, 2025. URL https://arxiv.org/abs/2512.13961.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V. Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=1PL1NIMMrw.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=gEZrGCozdqR.

A Related Work

Neural thickets and structured expert search. Gan & Isola (2026) propose that pretrained weights are surrounded by a dense population of task experts, motivating random perturbation, selection, and ensembling. Subsequent work examines whether observed candidate scores can guide a more efficient search. For instance, Lžičař (2026) scores an initial batch of perturbations, uses those scores to identify promising directions, and restricts further search to combinations of those directions. Their Qwen-7B Countdown audit finds no consistent advantage of learned directions over search controls, with results sensitive to candidate selection and the held-out questions. In a complementary approach, Cendra et al. (2026) use a probabilistic model of observed rewards to choose successive perturbations within a fixed set of random directions. On Qwen2.5-Instruct models from 0.5B to 3B, their preliminary results show that 200 evaluations often match or exceed 1,000-draw RandOpt for single-expert test accuracy. They also document cases where further selection gains yield flat or declining test accuracy. Our study complements these search-focused investigations by quantifying selection noise, measuring transfer across model families and sizes, and comparing selected-expert voting with repeated sampling from the anchor.

Random search and low-dimensional adaptation. Random weight guessing already provides a useful baseline on some control benchmarks (Oller et al., 2020). Evolution strategies turn such evaluations into an iterative learning procedure, using reward-weighted perturbations to update the model (Salimans et al., 2017; Qiu et al., 2026). Meanwhile, intrinsic-dimension studies and low-rank adaptation show that effective fine-tuning can use far fewer trainable parameters than the full model (Aghajanyan et al., 2021; Hu et al., 2022). TinyLoRA extends this observation to reasoning, obtaining substantial gains by training extremely small adapters with reinforcement learning (Morris et al., 2026). More recently, Lu et al. (2026) report successful adaptation by learning a binary mask that zeros a small fraction of pretrained weights while leaving the retained values unchanged. Such learned adaptations demonstrate the flexibility of pretrained representations. How frequently an unguided perturbation discovers a transferable improvement remains a separate empirical question, which our selection and held-out comparisons address.

Ensembling and test-time computation. Parameter Ensembling by Perturbation (PEP) averages predictions from Gaussian weight perturbations and tunes their variance on validation data (Mehrtash et al., 2020). Gan & Isola (2026) identify PEP as a close precedent for RandOpt, whose added step is candidate selection. For language models, self-consistency instead samples multiple reasoning paths from a single checkpoint and aggregates their answers (Wang et al., 2023). Repeated sampling can also substantially increase the probability of obtaining at least one correct solution (Brown et al., 2024). Recent work by Ba et al. (2026) reports higher pass@K for models trained with evolution strategies than with GRPO in its evaluated settings. Such coverage measures whether any sampled answer is correct; majority voting additionally requires the aggregation rule to return a correct answer. Our matched-K comparisons therefore assess the additional value of selected weight perturbations when both RandOpt and the unperturbed anchor receive the same answer budget.

B Experimental Details

B.1 Computational details

Software. Our implementation builds directly upon commit `536df0` of the official `sunrainyg/RandOpt` repository of Gan & Isola (2026). Unless explicitly noted (e.g., in Appendix E), all our experiments use this code to generate weight perturbations. However, we deviate from the official repository in one important respect. After evaluating a perturbed model, we restore the anchor weights exactly to their original values. This differs from the official implementation, which subtracts the perturbation in bf16 arithmetic, leaving a small rounding residual that can accumulate across candidates. We ablate this difference in Appendix E and find it to have little effect.

Inference. We use greedy decoding for all experiments by default, in keeping with Gan & Isola. When stochastic decoding is used, we sample with temperature 0.6 and top-$p = 0.95$ (Holtzman et al., 2020). Countdown and GSM8K use a 1,024-token limit, while MBPP uses a 2,048-token limit. We use vLLM (Kwon et al., 2023) with version 0.16.0 and PyTorch 2.9.1+cu128.

RandOpt. In keeping with the official implementation of Gan & Isola, for a given ith perturbation of the anchor model θ, we draw σ_i i.i.d. from the uniform distribution over $\{0.5, 1, 2\} \times 10^{-3}$ and then draw ε_i i.i.d. from the standard multivariate Gaussian distribution.[4] The perturbed model is then $\theta_i = \theta + \sigma_i \varepsilon_i$. We evaluate $N = 5{,}000$ candidates for OLMo-3-7B-Instruct and Qwen2.5-3B/7B-Instruct, and a smaller $N = 1{,}000$ candidates for Qwen2.5-32B-Instruct. The top-K candidates are selected by their mean reward on the selection set. At test time, RandOpt aggregates the top-K candidates by majority vote over valid answers.

Data. By default, we use the first $m = 200$ questions as the selection set for Countdown, GSM8K, and MBPP. For Countdown, we use the following $1{,}400$ questions as the held-out test set. For GSM8K and MBPP, we use the official test sets of $1{,}319$ and 500 questions, respectively. The separate Figure 1 (left) diagnostic uses 1,000 perturbations at $\sigma = 10^{-3}$, splitting Countdown's test set into 700 selection questions and 700 held-out questions. The top-right panel uses another set of 1,000 perturbations and samples random m-subsets from the $1{,}600$-question Countdown pool.

B.2 Scoring and voting

We preserve the scoring rules of Gan & Isola, which combine accuracy and formatting for Countdown, and use binary task rewards for GSM8K and MBPP. For Countdown, the selection reward decomposes as a sum of answer correctness r_{ans} and a format score r_{fmt}. In particular, the selection reward is $r_{\text{sel}} = r_{\text{ans}} + 0.1 r_{\text{fmt}}$. We often report test accuracy, which is simply r_{ans}. GSM8K simply records correctness of a numerical answer, while MBPP requires the produced code solution to pass all supplied tests. In all cases, malformed and unparsed outputs are counted as failures. We examine scoring sensitivity in Appendix E.4. For voting, we use the same procedure for both RandOpt and TT-MV of the anchor model. In particular, we exclude invalid answers before majority voting, and break ties in favor of the earliest voter (i.e., higher selection rank for RandOpt).

B.3 Uncertainty and density estimation

Uncertainty. Our selection-subset intervals report the 2.5th and 97.5th percentiles over 200 uniformly sampled selection sets, drawn without replacement while holding candidate outputs fixed. (Thus, the full-pool endpoint has only one distinct subset and lacks such intervals.) Paired-question 95% bootstrap intervals use 2,000 joint resamples of expert and anchor outcomes (Dror et al., 2018), conditional on the recorded candidates and voter orderings. Repeated answers remain grouped by question.

Accuracy density. Figures 6 and 7 compare $\sigma = 0.001$ perturbations with greedy decoding (Table 8) against 100 sampled anchor answer sets at $T = 0.6$, top-$p = 0.95$. Density is the fraction exceeding the specified multiple of each population's own mean accuracy on the same questions, with half credit for ties (tolerance 10^{-12}). We average over 200 matched random question orderings, taking prefixes of length m from the primary test sets; the model-size comparison fixes $m = 200$.

Model	Countdown	GSM8K	MBPP
OLMo-3-7B-Instruct	338	1,717	1,717
Qwen2.5-0.5B-Instruct	100	100	100
Qwen2.5-1.5B-Instruct	100	100	100
Qwen2.5-3B-Instruct	1,717	1,717	1,717
Qwen2.5-7B-Instruct	1,717	1,717	1,717
Qwen2.5-32B-Instruct	100	338	338

Table 8: Unselected perturbation counts in the fixed-scale accuracy-density analysis. Every model–task setting also has 100 stochastic anchor answer sets. These populations differ from the selected experts and the fresh Figure 1 scatter population.

[4] We note that this differs slightly from the σ range listed in Gan & Isola (2026), which instead uses $\{1, 2, 3\} \times 10^{-3}$. We follow the released implementation, which uses the $\{0.5, 1, 2\} \times 10^{-3}$ range.

C THE SELECTION-NOISE NULL MODEL

Under the selection-noise null model we consider, candidates share a reward distribution on each question and differ only through independent evaluation noise. (Intuitively, it is as if each candidate were simply a single rollout of a stochastic anchor model.) More precisely, for candidate i and question j, the model assumes that Y_{ij}, the candidate's reward on that question, takes the form

$$Y_{ij} = \mu_j + \varepsilon_{ij}.$$

Here μ_j is the common expected reward on question j, and ε_{ij} is a zero-mean random variable with variance v_j. The model assumes the errors ε_{ij} to be independent across candidate–question pairs.

Averaging m questions gives score variance v/m, where $v = m^{-1}\sum_{j\in S_m} v_j$. Consequently, the Gaussian approximation in Section 2.3 predicts $\mathbb{E}[\Delta_{\text{sel}}(m)] \approx a_N\sqrt{v/m}$. Note that the search size fixes a_N; for our experiments with $N = 1{,}000$ and $N = 5{,}000$, we have $a_{1000} \approx 3.241$ and $a_{5000} \approx 3.678$. However, v is data-dependent and must be estimated from the observed candidate rewards. (It is the only parameter of the Gaussian approximation that must be estimated.)

We estimate v directly from the candidate rewards. For a question pool of size Q, one simply has

$$\widehat{\mu}_j = \frac{1}{N}\sum_{i=1}^{N} Y_{ij}, \qquad \widehat{v} = \frac{1}{Q}\sum_{j=1}^{Q}\frac{1}{N}\sum_{i=1}^{N}(Y_{ij} - \widehat{\mu}_j)^2. \tag{5}$$

The divisor N gives the variance of the empirical distribution used for resampling. For uniformly sampled question subsets, this pool average is the expected within-subset variance. Binary rewards give $\widehat{v}_j = \widehat{p}_j(1-\widehat{p}_j)$. Countdown instead uses its recorded correctness-plus-format reward. We report the resulting $\widehat{v}$ values in Table 9 for every model–task setting. We note that the null curves in Fig. 1 (top right) and Fig. 2 use 120 empirical-resampling simulations per selection-set size.

Model	Task / analysis	Candidates N	Questions Q	$\widehat{v}$
OLMo 7B	Countdown, headline sweep	1,000	1,600	0.091526
	Countdown, transfer diagnostic	1,000	1,000	0.090403
	GSM8K	5,000	1,000	0.027554
	MBPP	5,000	400	0.100409
Qwen 3B	Countdown	5,000	1,000	0.067737
	GSM8K	5,000	1,000	0.081733
	MBPP	5,000	400	0.109001
Qwen 7B	Countdown	5,000	1,000	0.135793
	GSM8K	5,000	1,000	0.042690
	MBPP	5,000	400	0.068317
Qwen 32B	Countdown	1,000	200	0.127471
	GSM8K	1,000	1,000	0.032726
	MBPP	1,000	400	0.040870

Table 9: Mean within-question reward variance in every selection-noise comparison. Values use raw reward-squared units, before multiplying lifts by 100 to express points. All rows use the original three-scale candidate populations. Qwen-32B Countdown has a fixed 200-question comparison rather than a selection-size sweep.

D ADDITIONAL EXPERIMENTAL RESULTS

D.1 SELECTION AND TRANSFER

Figure 1 (top right) averages 200 uniform question subsets drawn without replacement from the 1,600-question Countdown pool, for each $m \in \{25, 50, 100, 200, 400, 800\}$. The full-pool endpoint uses the complete 1,600 questions once. We include complementary OLMo selection-and-transfer sweeps in Table 10, comparing observed selection reward lift, the noise-null prediction, and held-out accuracy gains. Selection lift falls as m grows, while held-out gains over the perturbation population remain near zero on GSM8K and modest on Countdown and MBPP.

Task	m	Apparent lift	Noise null	Held-out rank-1
Countdown ($N = 1000$)	25	18.40	18.03 ± 3.07	$0.67\ [-1.92, 3.09]$
	50	13.37	12.89 ± 1.66	$0.76\ [-2.09, 2.92]$
	100	10.08	9.38 ± 1.09	$1.05\ [-1.09, 3.42]$
	200	7.69	6.73 ± 0.68	$1.04\ [-0.92, 3.42]$
	400	5.98	4.78 ± 0.52	$1.07\ [-0.58, 3.42]$
	800	4.80	3.42 ± 0.34	$0.65\ [-0.41, 2.61]$
GSM8K ($N = 5000$)	25	9.63	8.57 ± 3.34	$-0.20\ [-1.73, 0.93]$
	50	7.60	7.61 ± 1.71	$-0.06\ [-1.35, 1.16]$
	100	5.90	5.68 ± 0.95	$0.00\ [-1.27, 1.16]$
	200	4.58	4.06 ± 0.50	$0.09\ [-1.27, 1.16]$
	400	3.50	2.96 ± 0.32	$0.15\ [-1.27, 1.16]$
	800	2.72	2.10 ± 0.16	$0.48\ [-0.44, 1.16]$
MBPP ($N = 5000$)	25	20.56	20.77 ± 3.28	$0.43\ [-3.09, 3.12]$
	50	15.59	15.32 ± 1.62	$0.94\ [-1.88, 3.72]$
	100	11.86	11.06 ± 0.94	$1.18\ [-1.88, 3.72]$
	200	8.75	8.03 ± 0.77	$1.30\ [-1.88, 3.72]$
	400	7.30	5.68 ± 0.46	1.72

Table 10: OLMo selection-and-transfer sweeps. Lift and null entries are reward points above the perturbation population; null entries give mean ± simulation SD. Held-out entries are accuracy points with selection-subset percentile intervals.

Model	Task	$m = 25$	$m = 200$	$m = 400$	$m = 800$
Qwen 3B	GSM8K	14.97	9.11	7.77	6.87
	MBPP	21.74	10.12	7.92	—
Qwen 7B	GSM8K	9.04	5.65	4.75	3.98
	MBPP	16.65	6.87	5.51	—
Qwen 32B	GSM8K	7.01	4.31	3.51	3.00
	MBPP	9.44	4.38	3.37	—

Table 11: Qwen2.5-Instruct rank-1 selection reward lift over the perturbation population (points) as selection-set size m increases. Searches use $N = 5,000$ candidates for 3B/7B and $N = 1,000$ for 32B. Entries average 200 uniform question subsets drawn without replacement from 1,000 GSM8K questions or 400 MBPP tasks; the MBPP $m = 400$ endpoint uses the full pool once (and $m = 800$ is unavailable).

Sensitivity to the selection questions. Naturally, individual OLMo experts are more sensitive to the selection questions than group averages. At $m = 200$ on Countdown, the selection-subset interval for held-out accuracy gain over the perturbation population is $[-0.92, 3.42]$ points for rank 1, versus $[0.51, 1.49]$ for the top-20 mean, despite average gains of about one point in both cases. The top-50 intervals remain close to zero on GSM8K ($[-0.20, 0.19]$) and slightly positive on MBPP ($[0.83, 1.58]$).

Qwen-3B Countdown retains a larger advantage (Table 12): increasing m from 200 to 800 barely changes the observed selection lift (24.08 to 23.83 points), while the null falls from 7.41 ± 0.81 to 3.56 ± 0.32. At $m = 200$, its top-50 held-out gain is 4.98 accuracy points above the perturbation population, with a selection-subset interval of $[4.33, 5.63]$. This strongly suggests a more persistent performance increase among top candidates.

D.2 Individual experts and anchor variability

Individual-expert comparisons. Figure 3 compares the selected experts with five-run anchor means (Table 19), retaining their original ranks and test outputs. Table 13 preserves the earlier matched comparisons and their paired-question intervals.

Selection and holdout. Figure 1 (left) uses 1,000 fresh $\sigma = 0.001$ perturbations and a separate 700/700 split drawn uniformly at random (seed 42), with selection–holdout accuracy correlation

Model	Task	Lift	Noise null	Null / lift	Top-50 transfer
OLMo 7B	Countdown	7.69	6.73	87.6%	0.93
	GSM8K	4.58	4.06	88.6%	0.00
	MBPP	8.75	8.03	91.8%	1.18
Qwen 3B	Countdown	24.08	7.41	30.8%	4.98
	GSM8K	9.11	6.90	75.8%	3.90
	MBPP	10.12	8.38	82.7%	2.83
Qwen 7B	Countdown	18.16	9.55	52.6%	6.40
	GSM8K	5.65	4.85	85.9%	2.07
	MBPP	6.87	6.69	97.5%	0.86
Qwen 32B	Countdown	8.55	7.96	93.1%	—
	GSM8K	4.31	3.66	85.1%	1.08
	MBPP	4.38	4.38	100.1%	0.25

Table 12: Selection-noise comparisons at $m = 200$. Lift and null use reward points; top-50 transfer averages individual held-out accuracy gains over the perturbation population. Entries average 200 selection subsets, except Qwen-32B Countdown, which uses the fixed first 200 questions and lacks a population-wide holdout comparison. Countdown transfer uses the auxiliary 600-question holdout; other tasks use their primary tests. OLMo Countdown and Qwen-32B use $N = 1000$; other settings use $N = 5000$.

Model	Task	N	Rank 1 / base	Difference [95% interval]
OLMo 7B	GSM8K	5,000	85.44/87.41	-1.97 $[-3.15, -0.76]$
OLMo 7B	MBPP	5,000	63.80/61.60	$+2.20$ $[-1.40, 5.80]$
Qwen 3B	GSM8K	5,000	81.12/80.67	$+0.45$ $[-1.52, 2.28]$
Qwen 3B	MBPP	5,000	56.00/56.60	-0.60 $[-4.01, 3.40]$
Qwen 3B	Countdown	5,000	26.79/15.43	$+11.36$ $[8.93, 14.07]$
Qwen 7B	GSM8K	5,000	87.95/88.02	-0.08 $[-1.36, 1.14]$
Qwen 7B	MBPP	5,000	63.20/65.00	-1.80 $[-4.80, 1.20]$
Qwen 7B	Countdown	5,000	43.71/33.36	$+10.36$ $[7.64, 13.07]$
Qwen 32B	GSM8K	1,000	91.51/90.07	$+1.44$ $[0.15, 2.73]$
Qwen 32B	MBPP	1,000	77.60/79.90	-2.30 $[-5.10, 0.30]$
Qwen 32B	Countdown	1,000	48.21/47.29	$+0.93$ $[-1.43, 3.14]$

Table 13: Historical matched rank-1 solo performance on the same questions as the anchor. Countdown uses the common 1,400-question test and MBPP the official 500-question test. The anchor column averages the recorded executions within each comparison. Differences are accuracy points, with paired-question 95% bootstrap intervals conditional on the saved executions. Figure 3 instead uses the five-run means in Table 19.

$r = 0.24$. Its rank-1 expert scores 69.86% on selection and 69.00% on holdout; the top-20 holdout mean is 68.68%. The five-run anchor averages 69.06% on holdout (range 68.43–69.71%), compared with 66.74% on selection. Thus the equally sized halves have different baseline accuracies. The markers follow the 700-question selection ranking.

Repeated execution of one expert. To measure variability with fixed weights, we reconstruct candidate 930 at $\sigma = 5 \times 10^{-4}$ from a clean bf16 anchor and evaluate both conditions greedily five times with generation seeds 42–46 on the same 1,400 Countdown questions. The expert averages 0.20 points below the anchor, with a 95% interval of $[-1.63, 1.23]$ (Table 14). We obtain this interval by averaging the five paired differences within each question, then bootstrapping questions 5,000 times.

Across the ten pairs of executions, mean final-answer disagreement is 6.06% for the anchor and 6.00% for the expert; correctness disagreement is 5.81% and 5.89%. These rates compare extracted numerical answers, treating invalid responses as empty, rather than comparing the full generated text.

Generation seed	Anchor	Candidate 930	Difference
42	68.07	67.93	−0.14
43	67.86	67.79	−0.07
44	68.14	67.64	−0.50
45	68.07	67.57	−0.50
46	67.36	67.57	+0.21
Mean	67.90	67.70	−0.20

Table 14: Matched OLMo Countdown repeats on the same 1,400 questions. Accuracy is in percent; differences are candidate 930 minus anchor, in points.

D.3 Validity and correctness on Countdown

This decomposition uses Qwen-3B, 7B, and 32B on the primary 1,400-question test, with both candidate and anchor outcomes aligned to source rows 200–1599. The marginal decomposition below retains the historical anchor executions. The shared-valid-answer analysis in Appendix D.3.1 instead uses all five fresh anchor runs, matching the updated comparisons in Section 3.2.

Countdown answers can fail because no valid equation is extracted or because a valid equation misses the target. To distinguish these sources of improvement across the three Qwen sizes, write accuracy as $c = vg$, where v is the probability of an extractable valid equation and g is correctness conditional on validity. Subtracting the anchor's accuracy gives

$$\Delta c = (\Delta v)g_0 + v_0(\Delta g) + (\Delta v)(\Delta g). \tag{6}$$

For a group of experts, we pool their candidate–question outcomes before taking $g = \bar{c}/\bar{v}$; this is not the unweighted mean of each expert's conditional success rate. The decomposition concerns binary test accuracy, without the $0.1r_{\text{fmt}}$ selection bonus. The validity flag checks extraction and the scorer's expression requirements, including use of the supplied numbers. Without the underlying rejected responses, it cannot distinguish tagging failures from every other reason an equation was rejected.

Group	Total gain	Validity	Conditional correctness	Interaction
Qwen-3B ($N = 5000$)				
Population	−2.01	−0.15	−1.88	+0.02
Top 20	+4.32	+4.15	+0.13	+0.04
Top 50	+3.32	+3.79	−0.37	−0.09
Selected rank 1	+11.36	+6.91	+3.07	+1.38
Qwen-7B ($N = 5000$)				
Population	−1.00	−1.53	+0.56	−0.03
Top 20	+6.18	+4.08	+1.87	+0.23
Top 50	+5.23	+3.25	+1.80	+0.18
Selected rank 1	+10.36	+3.57	+6.13	+0.66
Qwen-32B ($N = 1000$; selected candidates only)				
Top 20	−0.05	−0.59	+0.55	−0.01
Top 50	−0.56	−1.08	+0.53	−0.01
Selected rank 1	+0.93	−2.04	+3.10	−0.13

Table 15: Qwen Countdown primary 1,400-question test gain decomposition, in accuracy points. Top-K rows average individual experts, not their majority vote.

Qwen-3B's anchor has $v_0 = 26.14\%$ and $g_0 = 59.02\%$. For its top 20, $v = 33.18\%$ and $g = 59.52\%$: the baseline-weighted validity component is 4.15 points, approximately 96% of the total gain. For the top 50, validity rises to 32.56% while conditional correctness falls to 57.60%; the 3.79-point validity contribution exceeds the 3.32-point net gain because the other components are negative. Rank 1 has a different balance: $v = 37.86\%$ and $g = 70.75\%$, so validity contributes about 61% of its gain, with positive conditional-correctness and interaction terms.

The Qwen-7B anchor has $v_0 = 48.71\%$ and $g_0 = 68.48\%$. Among the top 50, validity rises to 53.46% and conditional correctness to 72.18%, so most of the group gain comes from producing more

Model	Both valid	Expert wins / losses	Shared-set gain	Valid-set change	Per-run p range
Qwen 3B	177.8	31.0 / 20.6	+0.74	+11.41	0.025–0.401
Qwen 7B	481.4	61.0 / 34.2	+1.91	+9.50	0.002–0.045
Qwen 32B	571.4	25.0 / 12.0	+0.93	+0.81	0.024–0.090

Table 16: Rank-1 Countdown gain split by shared versus changed valid-answer sets. Counts and gains average comparisons against five fresh greedy anchor runs. Both gain columns use accuracy points over all 1,400 questions and sum, before rounding, to the expert–anchor gap. Wins/losses count discordant correctness on shared valid-answer questions. The final column gives the range of unadjusted two-sided exact McNemar p-values computed separately for each run, not a test pooling repeated questions.

valid answers. For rank 1, validity reaches 53.93% and conditional correctness 81.06%; improved correctness accounts for the larger share of this candidate's gain. Changing which responses are valid also prevents a causal interpretation of the conditional-correctness component as arithmetic ability alone. At 32B, the anchor has $v_0 = 56.43\%$ and $g_0 = 83.80\%$. Its rank-1 expert has lower validity (54.00%) and higher conditional correctness (89.29%), leaving a small net gain. The top 20 and top 50 average 0.05 and 0.56 points below the anchor.

D.3.1 Comparisons on the same valid-answer questions

The marginal decomposition above compares conditional accuracies on different sets of valid responses. To compare the rank-1 expert and anchor on the same questions, let B denote questions where both produce valid expressions, E those where only the expert does, and A those where only the anchor does. Their total excess number of correct answers is exactly

$$n\Delta c = \sum_{j \in B} (c_{ij} - c_{0j}) + \sum_{j \in E} c_{ij} - \sum_{j \in A} c_{0j}. \tag{7}$$

The first term compares correctness on shared valid-answer questions. The other terms count correct answers gained and lost when the set of valid responses changes. All three terms use the original primary test and the original selection ranking. We compute the partition separately against each of the five fresh anchor executions and average its components. The resulting total equals the expert's accuracy minus the five-run anchor mean used in Figure 3.

Averaging over the five anchor comparisons, Qwen-3B produces valid expressions on 352.2 questions where the anchor does not, answering 233.6 correctly, and loses validity on 171.6 questions, 73.8 of which the anchor answered correctly. These changes contribute $100(233.6 - 73.8)/1400 = 11.41$ accuracy points, or 93.9% of its 12.16-point gain. The shared set contributes the remaining 0.74 points (31.0 wins, 20.6 losses). For Qwen-7B, changing the valid-answer set contributes 9.50 points, or 83.2% of its 11.41-point gain; shared-set correctness contributes 1.91 points. Qwen-32B gets 0.93 of its 1.74-point gain on the shared set. Qwen-7B's shared-set advantage is significant at the unadjusted 5% level against each anchor execution; Qwen-3B's and Qwen-32B's are significant against one and three of the five, respectively. These tests share the same expert and questions and do not constitute five independent replications. These are descriptive partitions of the observed gains: validity can fail for reasons beyond formatting, and conditioning on both outputs being valid does not identify a causal change in general reasoning ability.

E Implementation Audit Details

E.1 Cross-tensor coupling

The released sampler resets the random generator to the candidate's seed before perturbing each tensor. As a result, tensors with the same shape and dtype receive identical noise arrays. OLMo's 259 exposed tensors fall into six shape classes, so substantial sharing occurs across layers, while each array still contains many random coordinates. In our OLMo Countdown control, removing this coupling produces similar behavioral diversity: the pairwise correctness-disagreement rates are

13.33%, 14.62%, and 16.86% across the three perturbation scales with tied noise, versus 13.46%, 14.50%, and 16.39% with untied noise. Thus, this particular implementation choice has little effect on the measured disagreement rates.

E.2 FINITE-PRECISION RESTORATION

The reference implementation attempts to restore the anchor by subtracting each perturbation after evaluation. Because bf16 addition and subtraction incur rounding, $(w \oplus \sigma\varepsilon) \ominus \sigma\varepsilon$ can differ from w; successive candidates can therefore accumulate weight drift. To assess its effect, we compare this procedure with copying the stored anchor weights back after every candidate.

The two $N = 5,000$ OLMo Countdown searches use the same candidate seeds, perturbation scales, and worker assignments, with 1,250 candidates per worker across four inference workers. The released and exact-restoration procedures select candidates 930 and 2948, respectively; both answer 152 of the 200 selection questions correctly. These are distinct selected models from a shared perturbation population. Table 17 compares the selection statistics and individual experts.

Quantity	Drift-on	Exact restore
Selected candidate	930	2948
Selection reward, $m = 200$ (points)		
Population mean	74.027	73.984
Across-candidate SD	3.436	3.424
Rank-1 reward	82.840	82.715
Top-50 mean lift over population	6.549	6.428
Held-out accuracy (%)		
Rank-1 expert	67.71	68.14
Matched anchor	67.14	68.14
Rank-1 minus anchor (points, 95% interval)	$+0.57$ $[-1.36, 2.43]$	0.00 $[-1.86, 1.86]$

Table 17: OLMo-3-7B-Instruct Countdown restoration control, with $N = 5,000$ candidates per search. Selection statistics multiply raw rewards by 100; SD is across candidate mean rewards. Test accuracies use the same 1,400 questions. The final row gives paired-question 95% bootstrap intervals conditional on the saved executions, using the original matched anchors rather than the later five-run baselines.

Despite the change in selected candidate, observed single-expert gains remain below one accuracy point in both configurations. At the population level, exact restoration changes mean selection reward by only 0.043 points and the top-50 selection lift by 0.12 points. The held-out voting curves likewise remain close: their largest separation across the evaluated budgets is 0.50 accuracy points, falling to 0.07 at $K = 50$ (Table 18).

K	Anchor TT-MV	RandOpt drift-on	RandOpt exact restore
1	67.02	67.71	68.14
2	74.84	74.57	74.57
3	78.47	77.21	77.29
5	81.81	80.14	80.29
10	85.71	84.14	83.93
20	89.02	87.79	87.86
30	90.37	89.64	89.50
40	91.25	90.50	90.43
50	91.64	90.93	90.86

Table 18: OLMo Countdown vote curves. Figure 1 (bottom right) shows the anchor and released restoration (drift-on) columns on the common 1,400-question test. Anchor values use the stored 50-sample bank during figure generation. The exact-restoration control uses the same candidate seed list.

The TT-MV value at $K = 1$ averages all 50 stored samples, giving 67.02%. For $K = 3$ through 40, we average 20 random voter orderings with seed 20260908; at $K = 50$, we use the entire bank in its

Model	Task	Mean	Minimum	Maximum	SD
OLMo 7B	Countdown	67.27	66.71	68.21	0.57
	GSM8K	87.48	87.26	87.64	0.16
	MBPP	61.72	60.40	62.60	0.88
Qwen 3B	Countdown	14.63	14.36	15.00	0.24
	GSM8K	80.55	80.29	81.12	0.36
	MBPP	55.48	55.40	55.80	0.18
Qwen 7B	Countdown	32.30	32.07	32.50	0.17
	GSM8K	87.57	87.34	87.79	0.19
	MBPP	65.08	64.80	65.60	0.33
Qwen 32B	Countdown	46.47	46.29	46.79	0.19
	GSM8K	89.58	89.46	89.76	0.13
	MBPP	80.36	80.20	80.40	0.09

Table 19: Accuracy across five fresh greedy anchor executions, in percent. SD uses divisor four; the minimum and maximum describe the observed runs, not a confidence interval. Each execution evaluates 1,400 Countdown, 1,319 GSM8K, or 500 MBPP questions.

stored order. The added $K = 2$ point uses 20 separately seeded orderings, preserving all existing curve values. Voting retains the released valid-answer filtering and first-seen tie rule. All curves are recomputed after restricting the saved outputs to the common 1,400 test questions; no new model generations are involved. The full RandOpt curves continue to $K = 500$, but Wave 1 has only 50 anchor samples, so our matched comparisons stop at $K = 50$.

At $K = 50$, the RandOpt-minus-anchor gaps are -0.71 $[-1.79, 0.36]$ points for the released restoration arm and -0.79 $[-1.71, 0.21]$ for exact restoration. Both intervals include zero.

We also check whether candidate scores trend with the number of perturb–restore cycles, as would be expected if accumulated drift systematically improved or degraded performance. Regressing mean selection reward on cycle index separately for each worker gives slopes between -0.523 and $+0.832$ reward points per 1,000 cycles with drift, compared with -0.190 to $+0.516$ under exact restoration. Slopes vary in sign across workers in both runs. Together with the closely matched population and voting results, this control provides little evidence that restoration drift accounts for the missing large expert gain in this OLMo Countdown setting.

E.3 Decoder instability

Even with greedy decoding, repeatability depends on the execution configuration. Two anchor runs using batches of 200 produce identical final Countdown answers, despite possible differences in the generated text. By contrast, two evaluations that each generated 2,000 answers differ on 140 of the primary test questions: both answer and correctness disagreement reach 10.00%, while accuracy moves from 67.14% to 68.14%. We compute these comparisons from the saved outputs, preserving the original batching.

Repeating the same 1,000 exactly restored perturbations on 200 selection questions also reveals variation: 12.34% of answers and 12.30% of correctness flags change. However, those executions used different batch compositions, so their disagreement rates cannot isolate a difference in reproducibility between experts and the anchor. For that comparison, we use the matched fixed-candidate repeats in Appendix D.2.

Five-run anchor baselines. For every model–task pair in Figure 3, we collect five greedy executions in fresh processes with seeds 42–46, fixed question order and batching, and unchanged weights. Table 19 reports their means and observed variation. These runs use vLLM 0.16.0, PyTorch 2.9.1+cu128, and bf16; OLMo and Qwen-3B/7B run on RTX4090 GPUs with tensor-parallel size one, while Qwen-32B uses two H100 GPUs. Eager execution is enabled and prefix caching disabled. This bank supplies the Figure 3 baselines and is separate from the matched candidate-930 experiment in Appendix D.2.

E.4 SCORING SENSITIVITY AND EXECUTION VARIABILITY

Sensitivity to answer formatting. On GSM8K, even a trailing punctuation mark can prevent a correct answer from receiving credit. We therefore rescore the saved answers with a more tolerant comparator: after trimming whitespace and removing one trailing period or comma, a leading dollar sign, and thousands separators, we compare numerically whenever the prediction and reference both parse as numbers. Throughout this check, the extracted answers, expert rankings, winning vote identities, and original paired anchor executions remain fixed. Only the answer-acceptance rule changes.

Model	Anchor accuracy	Rank-1 gap	Top-50 enrichment	Voting gap ($K = 50$)
OLMo 7B	$87.41/92.46$	$-1.97/-2.24$	$-0.05/-0.13$	$+0.56/+0.25$
Qwen 3B	$80.67/85.52$	$+0.45/-0.61$	$+3.91/+1.26$	$-0.61/-0.47$
Qwen 7B	$88.02/91.81$	$-0.08/-0.08$	$+2.20/+0.69$	$-0.28/-0.64$
Qwen 32B	$90.07/95.75$	$+1.44/+0.61$	$+1.18/+0.01$	$+0.32/+0.11$

Table 20: GSM8K reference/tolerant scoring on the same 1,319 test questions. Anchor accuracy is in percent; the other entries are accuracy-point differences. Rank-1 and voting gaps are relative to the anchor; top-50 enrichment is relative to the perturbation population.

As Table 20 shows, this more lenient rule raises anchor accuracy and reduces some apparent expert gains. In particular, Qwen-3B's rank-1 gap changes sign, while Qwen-32B's top-50 enrichment falls from 1.18 to 0.01 points. The voting results, however, remain close: across all four models, the RandOpt–anchor gap at $K = 50$ stays below one point in magnitude under either convention. We retain reference scoring for the primary results; the alternative provides a sensitivity check.

Even within the reference implementation, selection reward and test accuracy need not agree. For example, in one GSM8K anchor execution over 2,319 questions, 86 answers pass the accuracy check yet receive zero selection reward. Countdown separates the two explicitly through its 0.1-weighted format bonus, which contributes approximately 0.83 points to the OLMo top-50 selection advantage.

Direct comparisons with the published results also depend on scorer version. After publication, a correction replaced the stored anchor reward with correctness and revised Countdown equation parsing. These changes further complicate reproduction of the original evaluation pipeline.

Execution variability in MBPP. Grading MBPP requires running generated programs, which may fail to terminate or allocate excessive memory. Execution safeguards are therefore necessary, and we count failed generations in the accuracy denominator.

We also found conflicting grades for outputs sharing the same program hash. Inspection of 53 such pairs, spanning seven test questions and two selection questions, confirmed that every pair contains byte-identical extracted code and correctly stored hashes. Thus hash collisions do not explain these discrepancies. The failing outputs also carry no flag indicating rejection by an execution safeguard.

For one affected program, changing Python's string-hash seed yields four passes in 20 clean executions: scoring-side randomness can indeed produce conflicting verdicts for identical code. The same test, however, did not reproduce the other conflicts: the checked programs on the other eight questions remain stable across these hash seeds. Worker state or execution timing may explain those discrepancies, but the available evidence does not identify the cause.

Despite these inconsistencies, none of the identified conflicts changes a recorded winning RandOpt or anchor vote at the evaluated budgets. We therefore retain the original grades. Assessing their effect on selection ranks would require broader rescoring through the production procedure.