
Budgeted Sequential Acquisition of Correlated Verifier Scores for Best-of- N Selection Without a Live Oracle

Han Shaun Lee
Stanford University
hanl@stanford.edu

Abstract

Weaver-style verifier ensembles select an answer among N sampled candidates by running every weak verifier on every candidate and aggregating; the verification bill grows linearly in the number of verifiers, and because weak verifiers make correlated errors, extra judges can add cost without adding signal. We study *budgeted sequential acquisition* for fixed-candidate best-of- N selection with no correctness oracle at test time: per question, decide which verifier to run next (over all N candidates) and when to stop. Our policy (DAS) values each unacquired verifier by the probability that its evidence would overturn the current selection times its non-redundant information given the verifiers already acquired, per unit cost, under a shrinkage Gaussian surrogate fit on development questions. On the released Weaver score tables for Llama-3.1-8B generations (MATH500, GPQA, MMLU-Pro; 13 reward/process-reward judges; $N=100$), with a hidden-score evaluator that masks unacquired scores and labels at the policy boundary, 10 question-level resplits and operating points frozen on a calibration split, DAS at its calibration-chosen operating point is within -0.014/-0.015/-0.006 of the full 13-judge label-assisted ensemble while using 3.6/8.7/9.6 of 13 calls per question. The savings are not due to adaptivity: a question-level pooled analysis with a pre-declared ± 0.02 margin shows DAS to be *equivalent* to cost-matched static dev-selected subsets and to confidence-only stopping in 13 of 18 comparisons, with no comparison favouring DAS by more than about one point. Under domain shift DAS is never worse than confidence-only and wins clearly on one of six pairs (MMLU-Pro $\rightarrow$ MATH500, +0.088), which a trace diagnostic attributes to DAS’s question-independent *first* judge (accuracy after one call 0.833 vs. 0.568) rather than to sequential decisions. Weaver’s binarised label model is the strongest full-ensemble selector on GPQA and MMLU-Pro (DAS trails it by -0.105/-0.039), and the released Weaver distilled cross-encoders, trained on 70B generations, do not transfer to these 8B candidates (0.527/0.347/0.539 Selection@1). Per-question adaptivity at judge-over- N granularity is therefore a careful negative. The falsifying experiment, candidate-level acquisition, comes out positive. A shortlist cascade runs one judge on all candidates and the remaining judges only on its top- m . It beats DAS by +0.014/+0.045/+0.040 at 139/149/185 instead of 214/584/671 candidate-judge evaluations per question (the full ensemble needs 1300). It matches the full ensembles except the label model on GPQA, and controls rule out leakage. Cached scores support evaluation-count claims only.

1 Introduction

Repeated sampling raises the chance that a language model produces a correct answer somewhere among N candidates far above its pass@1 [Brown et al., 2024, Snell et al., 2024]; the remaining problem is *selection*. Weaver [Saad-Falcon et al., 2025] closes much of this generation–verification gap by scoring each candidate with an ensemble of weak verifiers (reward models, process reward models, LM judges) and aggregating the scores with a weak-supervision label model whose parameters are estimated without correctness labels. Weaver then distils the ensemble into a 400M cross-encoder to make inference cheap.

The ensemble itself is expensive: every verifier is run on every candidate of every question, so verification compute scales as (verifiers) $\times N$. Two facts in Weaver’s own analysis suggest that much of this spend is avoidable. First, verifier errors are correlated—Weaver already has to drop or model dependent verifiers (use_deps) and prune low-signal ones—so the marginal value of a judge depends on which judges have already been consulted. Second, question difficulty varies: for many questions one strong judge already separates the candidates, while for others the ensemble disagrees and more evidence is needed. A fixed ensemble ignores both.

We study the natural adaptive alternative. For each question, with the candidate set fixed, a policy sequentially decides which verifier to acquire next (over all N candidates) and when to stop, under a per-question budget, *without a correctness oracle at test time*. Our contributions are:

- **Problem formulation and evaluator.** We formalise budgeted acquisition of correlated verifier scores for fixed-candidate best-of- N selection, define the cost unit (one candidate–judge evaluation; one acquisition is a judge over all N candidates), and build a hidden-score environment that masks unacquired scores and all labels at the policy boundary with a test proving the masking (Sec. 3).
- **A dependence-aware sequential policy (DAS).** A Gaussian surrogate over per-judge evidence gives, for every unacquired judge u and acquired set A , the conditional standard deviation $\sigma_{u|A}$ (how unpredictable u ’s evidence still is) and an information term; the value of u is the probability that its evidence would overturn the current top candidate times its new-information content, divided by cost (Sec. 4). The policy stops when no judge is worth its cost. We give label-assisted and unlabeled instantiations of both the dependence model and the selector.
- **A faithful baseline and a careful comparison.** We executed the upstream Weaver code on the released 8B tables (its released, transductive protocol) and validated a numpy reimplementation that reproduces it to within init noise, allowing Weaver to be run on arbitrary verifier subsets and on our frozen splits (Sec. 5). All comparisons are paired at the question level with bootstrap confidence intervals and a pre-declared equivalence margin, on three datasets and ten question-level resplits, in-domain and across domains.
- **Findings.** (i) Negative: acquiring a few well-chosen judges recovers most of the full ensemble’s accuracy (72%/34%/26% fewer calls at the best-cal point), but a static dev-selected subset of the same size does as well, so per-question judge-level adaptivity is statistically equivalent to static selection. Under shift, DAS’s one large win over confidence-only stopping is explained by its first-judge ranking rule. (ii) Mechanism: removing the dependence penalty, or adding synthetic copies of a judge, shows that the penalty is what avoids redundant judges. (iii) Baselines: Weaver’s label model is the best full-ensemble selector on GPQA and MMLU-Pro once a seed-dependent initialisation failure in our reimplementation is fixed, and the released distilled verifier fails under a 70B→8B generator change. (iv) Positive: candidate-level acquisition through a per-question shortlist cascade beats DAS with 1.5–4 $\times$ fewer evaluations and matches the full listwise ensemble at 11–14% of its cost.

We do not claim that verifier correlation, weighted voting, verifier routing, or online verification are new; Sec. 2 positions the work against Weaver’s dependency handling and the online multi-verifier verification of Hajishemi and Shen [2026].

2 Related work

Verifier ensembles and weak supervision. Weaver [Saad-Falcon et al., 2025] aggregates binarised verifier scores with a conditionally-independent label model [Ratner et al., 2016, 2019], es-

timates class balance and binarisation thresholds from a small labelled set, drops verifiers with extreme marginals, and offers dependency handling (`use_deps` $\in$ {none, drop, model}): drop searches for the maximally independent verifier subset using the inverse covariance with the truth appended, and model passes learned dependencies to the label model’s inverse-form estimator. Learning dependency structure among weak sources without labels is established [Bach et al., 2017], as is fast moment-based parameter estimation [Fu et al., 2020]. We therefore treat dependence modelling as part of the baseline, and our contribution is the *sequential, per-question* decision of which dependent source to pay for.

Online verification with an oracle. OMVV [Hajihashemi and Shen, 2026] studies a stream of (query, single response) pairs, routes each round to one of K weak verifiers with an exponential-weights policy under bandit feedback, applies accept/reject thresholds, and queries a costly ground-truth oracle in the uncertain region; oracle labels drive all its learning, and it proves error control and sublinear routing regret. Our setting differs in every structural respect: a fixed set of N candidates per question, best-of- N *selection* rather than accept/reject, no oracle at test time, and acquisition of *several* correlated judges per question that must be aggregated. The formulations do not coincide; we reuse OMVV’s idea of exploiting generator self-agreement as a free feature (upstream Weaver’s `mv_verifier`) and find it harmful for selection with an 8B generator (Sec. 4.1).

Cascades, routing and value of information. Model cascades and routers [Varshney and Baral, 2022, Chen et al., 2023] choose one model per query; multi-agent verification [Lifshitz et al., 2025] runs all verifiers. Our acquisition rule is a myopic value-of-information heuristic [Howard, 1966] with a conditional-variance redundancy penalty in the spirit of Gaussian-process sensor placement [Krause et al., 2008]; the stopping statistic is a probability-of-flip criterion related to selective prediction [Geifman and El-Yaniv, 2017].

3 Problem setup and evaluation protocol

Data. We use the released Weaver score tables for Llama-3.1-8B-Instruct generations¹ on MATH500 (500 questions), GPQA (646 rows, 638 distinct questions) and MMLU-Pro (500). Each row holds $N=100$ sampled responses, per-response correctness labels, and 19 verifier score columns common to the three tables: 10 reward models and 3 process reward models each exposed as min/max/avg aggregates of one step-scoring pass. We inspected the parquet schema rather than trusting the README: scores are already in $[0, 1]$ with no missing values; GPQA/MMLU-Pro also carry a `weaver_scores` column (an upstream *output*, excluded as input); MATH500 carries an `internlm_scores` column absent elsewhere (excluded for a common pool); the HF repository referenced by upstream `constants.py` for MATH500-8B differs in name from the collection’s but is byte-identical in scores and labels. There is no cross-dataset question overlap; the 8 duplicated GPQA instructions are treated as one unit.

Acquisition unit and cost. A *judge* is a verifier model; a PRM’s three aggregates come from one call and form one judge with three features, giving $J=13$ judges. Acquiring judge u for a question runs it on all N candidates and costs $c_u N$ candidate–judge evaluations; under the primary *equal* cost model $c_u=1$, so the full ensemble costs 1300 units per question and “calls per question” counts judges. Because the data are cached, all efficiency claims are *call counts*; we report a declared parameter-count cost model as a sensitivity analysis and make no latency, dollar or energy claims.

Hidden-score environment. All scores and labels are physically present, so masking is enforced at the policy boundary: a policy receives an observation whose $(N \times 19)$ score array is NaN for every feature of every unacquired judge, together with the acquired mask, spent and remaining budget and the cost vector—and nothing else. Labels are read only by the evaluator after the policy commits a selection. `tests/test_masking.py` verifies that observations are read-only and label-free, that budgets are enforced, that a “spy” policy’s selections and acquisition orders are unchanged when all unacquired scores are randomised and all labels flipped (while its correctness flips), and that every aggregator’s output is invariant to hidden values.

¹HF collection `hazyresearch/weaver-683798010b39c9653ddb9bd8`; revisions and SHA-256 in `sources.lock.json`.

Splits and protocol. Questions (with all their responses and scores) are split at the question level into dev (40%), cal (20%) and test (40%) with ten seeds (three in the initial protocol, seven added later under the identical frozen protocol). Dev fits aggregator weights, judge reliabilities, dependence models and judge orders; cal fixes the listwise regulariser, the Weaver binarisation threshold, the class-balance scalar, and every sequential policy’s stopping threshold; test is evaluated once per frozen configuration. For domain shift, everything fitted and frozen on the source’s dev/cal is applied unchanged to the *entire* target table. Sequential policies are run once per question with an exhaustive budget while logging a replayable trace; since the greedy acquisition order does not depend on the stopping threshold, any threshold can be applied offline exactly.

Metrics. Selected-answer accuracy with question-level bootstrap CIs (paired for differences), regret relative to the oracle (any correct candidate), mean calls per question, calibration of a Platt-mapped confidence (ECE) and risk-coverage (AURC) using the stopping statistic as confidence, and accuracy under shift. Operating points: δ -rule (cheapest cal threshold whose cal accuracy is within δ of the full-ensemble cal accuracy with the same selector) and best-cal (cal-accuracy-maximising threshold); cost-matched comparisons interpolate static families between neighbouring k and choose the confidence-only threshold on cal to match DAS’s cal call count.

4 Method

4.1 Selectors (aggregators)

Given the acquired set A and masked scores, a selector maps each candidate i to a score $s_i(A)$; the committed answer is $\arg \max_i s_i$. Within-question standardisation z_{if} of each feature over the N candidates is available as soon as its judge is acquired. We use: (i) the **unweighted mean** of acquired raw scores (label-free; Weaver’s naive ensemble); (ii) a **listwise** (conditional-logit) model $s_i = \sum_{f \in A} w_f z_{if}$ whose weights maximise $\sum_q \log P(\text{select a correct candidate of } q)$ on dev, refit jointly for each acquired subset (lazily cached), with non-negative weights; and (iii) a **label-model selector**: Weaver’s binarised label model with parameters estimated from dev *scores only*, applied to acquired votes with unacquired judges abstaining. The non-negativity in (ii) was forced on us by a failure: unconstrained fits learn a *negative* weight on majority-vote agreement on GPQA/MMLU-Pro, because the plurality answer of an 8B generator is wrong about two thirds of the time and maximising probability mass on correct candidates pushes mass away from it—after which the $\arg \max$ lands on singleton answers (GPQA accuracy collapses to 0.21). We therefore exclude the agreement feature from the supervised selector and constrain weights to be votes.

4.2 Dependence model

Let $e_u(i) = r_u \bar{z}_u(i)$ be judge u ’s evidence for candidate i : its mean standardised feature times a reliability weight r_u (label-assisted: the univariate listwise slope of u alone; unlabeled: the label model’s estimated log-likelihood-ratio weight, or 1 for a fully label-free variant). On dev we fit a Gaussian surrogate over the J -vector of evidences with Ledoit–Wolf shrinkage [Ledoit and Wolf, 2004]. The *unlabeled* model uses the covariance of observed evidence—score dependence, a proxy for error dependence; the *label-assisted* model uses the pooled within-class covariance (error dependence in the sense of Bach et al. [2017]) and the joint covariance with the label. For acquired set A and candidate judge u ,

$$\sigma_{u|A}^2 = \Sigma_{uu} - \Sigma_{uA} \Sigma_{AA}^{-1} \Sigma_{Au}, \quad \text{info}(u|A) = \begin{cases} |\text{pcorr}(e_u, y \mid e_A)| & \text{label-assisted} \\ \sqrt{1 - R_{u|A}^2} \, r_u / \max_v r_v & \text{unlabeled.} \end{cases}$$

A judge whose evidence is predictable from A has small $\sigma_{u|A}$ and small info: this is the redundancy penalty.

4.3 Acquisition and stopping (DAS)

Let b be the surrogate belief $b_i = \sum_{u \in A} e_u(i)$ (unit-consistent with Σ), $\hat{i} = \arg \max_i b_i$ and gaps $g_k = b_{\hat{i}} - b_k$. If judge u were acquired, pairwise belief differences would move by $\mathcal{N}(0, 2\sigma_{u|A}^2)$

Algorithm 1 DAS for one question (policy sees only masked scores)

```
1:  $A \leftarrow \emptyset$ ; observe nothing
2: while some judge is affordable do
3:    $b \leftarrow \sum_{u \in A} e_u$ ; compute  $g_k$  from  $b$ 
4:   for each unacquired affordable  $u$  do compute  $\sigma_{u|A}$ ,  $\text{info}(u|A)$ ,  $P_{\text{flip}}(u|A)$ 
5:   end for
6:   if  $\max_u P_{\text{flip}}(u|A) \text{info}(u|A) < \varepsilon$  then break
7:   end if
8:   acquire  $u^* = \arg \max_u P_{\text{flip}} \text{info}/c_u$  over all  $N$  candidates;  $A \leftarrow A \cup \{u^*\}$ 
9: end while
10: return  $\arg \max_i s_i(A)$   $\triangleright$  selector of choice; labels never read
```

under the surrogate, so

$$P_{\text{flip}}(u|A) = \min \left(1, \sum_{k \neq i} \Phi \left(-g_k / (\sqrt{2} \sigma_{u|A}) \right) \right), \quad \text{value}(u) = \frac{P_{\text{flip}}(u|A) \text{info}(u|A)}{c_u}.$$

DAS acquires $\arg \max_u \text{value}(u)$ among affordable judges and stops when $\max_u P_{\text{flip}} \cdot \text{info} < \varepsilon$ or the budget is exhausted; the committed answer is $\arg \max_i s_i(A)$ under the chosen selector. ε is frozen on cal. Ablations remove the conditional variance (marginal σ_u), the info term, use diversity only ($\sqrt{1 - R_{u|A}^2}$), or add conditional-mean imputation of unacquired evidence to b .

5 Baselines

Upstream Weaver. We ran the released `selection/run.py` best configs (commit 5924c976, HEAD at fetch time; metal-ama 9da052c4) on CPU: MATH-500-8B 0.748, GPQA-Diamond-8B 0.465 (README reference 0.471), MMLU-Pro-8B 0.672 (`results/p04_upstream_logs`). The released protocol is transductive (the label model is fit on the evaluation set itself, `train_split=1.0`), uses that set’s labels for the class balance, includes a majority-vote pseudo-verifier, and uses dataset-specific binarisation thresholds (0.25/0.1/0.1) that were tuned by the authors on these same sets. Our numpy reimplemention (same overlap matrix, μ initialisation, loss and SGD schedule) reproduces 0.748–0.754 / 0.465 / 0.672–0.674 over five init seeds with identical kept-verifier counts (test in `tests/test_weaver_ref.py`), so we use it to run Weaver on any judge subset, with the class balance from cal and the threshold either cal-selected or upstream’s. **Others.** Dev-selected best single verifier; unweighted mean; static top- k by dev single-judge accuracy; static greedy forward selection on dev; static diverse subsets (best judge per correlation cluster); cost-matched random acquisition; confidence-only sequential stopping (fixed dev order, stop when the top-1/top-2 margin exceeds a cal-chosen threshold); the oracle. We also run the released $\sim 400\text{M}$ distilled Weaver cross-encoders on all $\sim 165\text{k}$ 8B candidates (Sec. 6).

6 Results

Reporting conventions. All results use 10 question-level resplits. In tables, $x \pm s$ is the mean over resplits $\pm$ its standard error across resplits (split variance). Because resplits share questions, our primary inference is a *question-level pooled* paired comparison (Table 2): each question’s paired difference is averaged over the resplits in which it was a test question, and we bootstrap over questions (10,000 resamples; GPQA duplicate instructions merged into one unit). Equivalence is a TOST at $\alpha=0.05$ with margin ± 0.02 accuracy, declared before the seven additional resplits were inspected: the 90% CI must lie inside the margin (marked $\approx$); * marks a 95% CI excluding zero. Sequential policies are reported at two operating points frozen on cal (the δ -rule with $\delta=0.01$ and best-cal). Test sets are touched once per frozen configuration.

The regime exists, and static selection already exploits it. Table 1 and Fig. 1 show the motivating failure: on MATH500 the unweighted mean of the three best dev judges reaches 0.831, but averaging all 13 falls to 0.739 (Fig. 4), and the single dev-selected best verifier (0.835) beats 13-judge Weaver. Oracle headroom is large everywhere (0.993/0.945/0.964). The gap between “a few

Table 1: In-domain held-out test accuracy (equal cost; calls per question out of 13; mean $\pm$ s.e. over 10 resplits). Oracle is diagnostic only. All label-model fits use upstream’s fixed initialisation seed (see text).

Method	calls/q	MATH500	GPQA	MMLU-Pro
Oracle (any correct; diagnostic)	–	0.993 $\pm$ 0.001	0.945 $\pm$ 0.004	0.964 $\pm$ 0.003
Majority vote (free)	0	0.684 $\pm$ 0.006	0.294 $\pm$ 0.008	0.571 $\pm$ 0.006
Best single verifier (dev-selected)	1	0.835 $\pm$ 0.004	0.367 $\pm$ 0.007	0.653 $\pm$ 0.008
Unweighted mean, all 13	13	0.739 $\pm$ 0.007	0.390 $\pm$ 0.005	0.691 $\pm$ 0.005
Weaver (upstream protocol threshold), all 13	13	0.749 $\pm$ 0.006	0.461 $\pm$ 0.009	0.671 $\pm$ 0.004
Weaver (cal threshold), all 13	13	0.786 $\pm$ 0.008	0.461 $\pm$ 0.009	0.664 $\pm$ 0.005
Label-model selector (dev-fit), all 13	13	0.803 $\pm$ 0.009	0.482 $\pm$ 0.005	0.722 $\pm$ 0.015
Listwise (label-assisted), all 13	13	0.859 $\pm$ 0.006	0.409 $\pm$ 0.007	0.698 $\pm$ 0.005
Static greedy subset (dev), $k=3$	3	0.847 $\pm$ 0.005	0.371 $\pm$ 0.006	0.698 $\pm$ 0.009
Static diverse subset (dev), $k=3$	3	0.833 $\pm$ 0.005	0.364 $\pm$ 0.014	0.670 $\pm$ 0.003
Random subset, $k=3$	3	0.688 $\pm$ 0.008	0.349 $\pm$ 0.007	0.649 $\pm$ 0.008
Confidence-only sequential ($\delta=0.01$)	4.2/7.1/6.2	0.839 $\pm$ 0.005	0.385 $\pm$ 0.009	0.675 $\pm$ 0.012
DAS (ours, label-assisted, $\delta=0.01$)	2.1/5.8/6.7	0.839 $\pm$ 0.006	0.384 $\pm$ 0.011	0.686 $\pm$ 0.005
DAS (ours, best-cal operating point)	3.7/8.6/9.6	0.849 $\pm$ 0.006	0.396 $\pm$ 0.008	0.692 $\pm$ 0.007
DAS label-free policy + mean selector ($\delta=0.01$)	2.4/2.1/4.5	0.776 $\pm$ 0.011	0.370 $\pm$ 0.009	0.662 $\pm$ 0.010
DAS + label-model selector (unlabeled weights, $\delta=0.01$)	4.2/7.8/10.2	0.799 $\pm$ 0.014	0.448 $\pm$ 0.008	0.700 $\pm$ 0.015

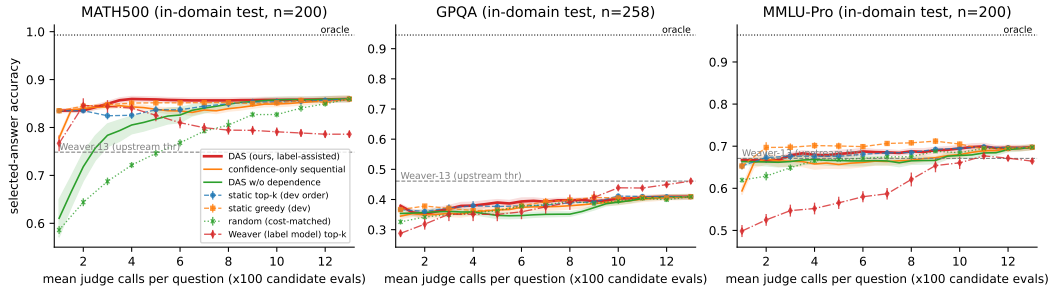


Figure 1: Accuracy versus mean judge calls per question on the in-domain test split (10 resplits; bands/bars are s.e. across resplits). Sequential curves sweep the stopping threshold on the test traces for visualisation only; reported operating points use cal-frozen thresholds.

good judges” and “all judges” is closed by *static* dev choices: a dev-greedy subset of three judges reaches 0.847/0.371/0.698 versus 0.859/0.409/0.698 for the full listwise ensemble.

DAS saves calls relative to the full ensemble, and is equivalent to static subsets. At the $\delta=0.01$ point DAS uses 2.1/5.8/6.7 of 13 calls at a pooled accuracy difference of -0.025 $[-0.040, -0.011]$, -0.023 $[-0.040, -0.004]$ and -0.012 $[-0.027, +0.003]$ relative to the full listwise ensemble (Table 2): the cal-level promise of $\delta=0.01$ does not transfer exactly, a 1–2.5 point shortfall that the noisy operating-point choice on 100 cal questions produces. At the best-cal point the shortfall shrinks to $-0.014/-0.015/-0.006$ at 3.6/8.7/9.6 calls. Against cost-matched alternatives the pooled analysis turns the earlier non-significance into positive evidence of *equivalence*: in 13 of the 18 comparisons with static top- k , dev-greedy and confidence-only stopping (3 datasets $\times$ 2 operating points) the 90% CI lies within ± 0.02 . The exceptions are small: DAS beats static top- k by $+0.013$ $[+0.004, +0.023]$ on MATH500 at best-cal and confidence-only by $+0.019$ $[+0.003, +0.035]$ on GPQA at $\delta=0.01$; the GPQA and MMLU-Pro comparisons with dev-greedy ($+0.012$ $[-0.007, +0.031]$, -0.022 $[-0.047, +0.001]$) and MMLU-Pro with confidence-only ($+0.008$ $[-0.010, +0.026]$) at $\delta=0.01$ are inconclusive. DAS exceeds cost-matched random acquisition by $+0.194$ on MATH500. Per-question adaptivity therefore buys at most about one point over a well-chosen static subset at judge-over- N granularity.

Selector choice matters more than acquisition. The best full-ensemble selector differs by dataset: listwise weights on MATH500 (0.859 vs. 0.803 for the label model), but Weaver’s binarised label model with dev-estimated, label-free parameters on GPQA (0.482 vs. 0.409) and MMLU-Pro

Table 2: Question-level pooled paired differences (DAS minus comparator; 95% bootstrap CI over questions; 10 resplits). $\approx$: equivalent within ± 0.02 (TOST, 90% CI); *: 95% CI excludes 0. Static families are interpolated to DAS’s mean calls; confidence-only uses a cal threshold matching DAS’s cal calls.

Dataset	point	DAS acc (calls)	full-13	static top-k	dev-greedy	conf.-only	best single	random	label model-13
MATH500	$\delta=0.01$	0.839 (2.1)	-0.025 [-0.040,-0.011]*	+0.008 [+0.003,+0.013]*	-0.000 [-0.013,+0.011]*	-0.003 [-0.022,+0.015]*	+0.003 [-0.018,+0.023]*	+0.194 [+0.166,+0.222]*	+0.034 [+0.006,+0.062]*
MATH500	best-cal	0.850 (3.6)	-0.014 [-0.024,-0.005]*	+0.013 [+0.004,+0.023]*	+0.006 [-0.007,+0.020]*	+0.003 [-0.017,+0.023]*	+0.014 [-0.012,+0.040]	+0.160 [+0.136,+0.185]*	+0.045 [+0.016,+0.074]*
GPQA	$\delta=0.01$	0.389 (5.8)	-0.023 [-0.040,-0.004]*	+0.003 [-0.011,+0.017]*	+0.012 [-0.007,+0.031]	+0.019 [+0.003,+0.035]*	+0.020 [-0.003,+0.043]	+0.020 [+0.002,+0.038]*	-0.105 [-0.137,-0.074]*
GPQA	best-cal	0.397 (8.7)	-0.015 [-0.026,-0.004]*	-0.002 [-0.014,+0.009]*	-0.005 [-0.022,+0.010]*	+0.007 [-0.006,+0.019]*	+0.027 [+0.001,+0.054]*	+0.015 [+0.001,+0.029]*	-0.098 [-0.130,-0.066]*
MMLU-Pro	$\delta=0.01$	0.680 (6.7)	-0.012 [-0.027,+0.003]	+0.000 [-0.017,+0.017]*	-0.022 [-0.047,+0.001]	+0.008 [-0.010,+0.026]	+0.036 [+0.008,+0.066]*	+0.014 [-0.004,+0.031]	-0.039 [-0.064,-0.015]*
MMLU-Pro	best-cal	0.687 (9.6)	-0.006 [-0.016,+0.004]*	-0.005 [-0.017,+0.006]*	-0.003 [-0.021,+0.013]*	-0.001 [-0.012,+0.011]*	+0.043 [+0.014,+0.073]*	+0.004 [-0.009,+0.017]*	-0.033 [-0.057,-0.009]*

Table 3: Per-resplit view of the paired in-domain differences for DAS ($\delta=0.01$); mean $\pm$ s.e. over resplits.

Source (in-domain test)	calls/q	Δ vs. full-13	vs. conf.-only	vs. static top-k	vs. greedy	vs. random
MATH500	2.1	-0.021 $\pm$ 0.006	-0.002 $\pm$ 0.006	+0.008 $\pm$ 0.005	-0.000 $\pm$ 0.004	+0.196 $\pm$ 0.016
GPQA	5.8	-0.025 $\pm$ 0.006	+0.018 $\pm$ 0.007	+0.001 $\pm$ 0.005	+0.013 $\pm$ 0.009	+0.021 $\pm$ 0.008
MMLU-Pro	6.7	-0.011 $\pm$ 0.003	+0.008 $\pm$ 0.009	-0.001 $\pm$ 0.007	-0.016 $\pm$ 0.006	+0.018 $\pm$ 0.008

(0.722 vs. 0.698). Label-assisted DAS therefore trails the 13-judge label model by -0.105 [-0.137,-0.074] on GPQA and -0.039 [-0.064,-0.015] on MMLU-Pro (Table 2, last column), while beating it on MATH500 (+0.034). Running DAS’s acquisition with the label-model selector gives 0.448 at 7.8 calls on GPQA and 0.700 at 10.2 on MMLU-Pro, recovering most of the label model’s advantage with modest savings. The fully label-free DAS (score-covariance dependence, mean selector) is weaker (0.776/0.370/0.662 at 2.4/2.1/4.5 calls): observable score correlation is useful for redundancy but carries no reliability information. *Label-model initialisation (a baseline bug we fixed)*. The label model initialises μ as μ_0 times one uniform draw. Our first runs seeded this draw with the resplit seed. For seeds 5, 7 and 9 the draws are 0.22, 0.076 and 0.010, and with upstream’s learning rate and 1000 epochs the fit never recovers: at seed 9, Weaver on MMLU-Pro fell below majority voting. Upstream’s configs fix the seed at 0, which gives a draw of 0.55. All label-model fits now use that fixed seed. A regression test (tests/test_weaver_ref.py) checks it, and all 30 runs were repeated; seed-0 runs are byte-identical to the originals. The fix raised 13-judge Weaver from 0.398 $\pm$ 0.031 to 0.461 $\pm$ 0.009 on GPQA and from 0.612 $\pm$ 0.051 to 0.671 $\pm$ 0.004 on MMLU-Pro (mean $\pm$ s.e. over resplits). Listwise and DAS results are unchanged.

The released distilled verifier does not transfer to 8B generations.

Weaver’s efficiency answer is a $\sim$ 400M ModernBERT cross-encoder distilled from the ensemble. We ran the released task-specific checkpoints (MIT; Weaver_Distilled_ModernBERT_Large_for_{MATH500,GPQA,MMLU-Pro}, upstream architecture and input format, strict state-dict load) on all 164,600 8B candidates on an L4 GPU and evaluated Selection@1 on exactly our test splits (Table 4). They reach 0.527/0.347/0.539, below the 13-judge listwise ensemble by -0.332/-0.063/-0.158 and below DAS at its $\delta=0.01$ point by -0.311/-0.037/-0.147; on MATH500 and MMLU-Pro they are even below free majority voting. Within-question AUC is 0.548/0.579/0.562 versus 0.836/0.578/0.713 for the best single judge. The checkpoints are not broken: on the first 10 questions of the 70B MATH500 table they were trained on, the MATH500 checkpoint has within-question AUC 0.814 (pilot). The failure is a generator shift (70B $\rightarrow$ 8B responses to the *same* questions; the checkpoints have seen these questions, which if anything favours them). Re-distilling on 8B pseudo-labels would be the fair comparison and is a disclosed gap (Sec. 8).

Domain shift: DAS is never worse than confidence-only, and the one large win is a ranking effect.

Table 5 applies everything frozen on a source’s dev/cal to the whole target table (pooled over the 10 source resplits; the bootstrap is over target questions and does not include dev-split variability, for which Table 6 gives the per-resplit spread). DAS beats cost-matched confidence-only stopping on MMLU-Pro $\rightarrow$ MATH500 (+0.088 [+0.071,+0.106]) and GPQA $\rightarrow$ MMLU-Pro (+0.019 [+0.008,+0.031]) and is equivalent on the other four pairs. (With three resplits, v1 of this paper reported small losses on five pairs; they do not survive more resplits.) The large win is not sequential adaptivity. With nothing acquired, every candidate gap is zero, so DAS’s first choice is the judge with the largest dev information term, the same judge for every question; for MMLU-Pro sources that is the math PRM QwenPRM, whereas the dev top-1-accuracy ranking used by static top- k and confidence-only picks a general reward model in 9 of 10 resplits. After a single call DAS is already

Table 4: Released Weaver distilled cross-encoders (one 0.4B call per candidate) on our in-domain test splits; paired differences mean $\pm$ s.e. over resplits, ($k/10$) resplits significant.

Eval set	distilled model	distilled acc	Δ vs. DAS ($\delta=0.01$)	Δ vs. DAS (best-cal)	Δ vs. full-13 listwise	Δ vs. label model-13
MATH500 test	task-specific	0.527 $\pm$ 0.008	-0.311 $\pm$ 0.008 (10/10)	-0.322 $\pm$ 0.006 (10/10)	-0.332 $\pm$ 0.006 (10/10)	-0.276 $\pm$ 0.006 (10/10)
GPQA test	task-specific	0.347 $\pm$ 0.009	-0.037 $\pm$ 0.012 (2/10)	-0.049 $\pm$ 0.008 (2/10)	-0.063 $\pm$ 0.009 (5/10)	-0.135 $\pm$ 0.008 (10/10)
MMLU-Pro test	task-specific	0.539 $\pm$ 0.006	-0.147 $\pm$ 0.006 (10/10)	-0.153 $\pm$ 0.008 (10/10)	-0.158 $\pm$ 0.006 (10/10)	-0.182 $\pm$ 0.014 (10/10)

Table 5: Cross-domain, question-level pooled paired differences for DAS ($\delta=0.01$, thresholds frozen on the source cal split).

Source $\rightarrow$ Target	DAS acc (calls)	vs. conf.-only	vs. static top- k	vs. full-13	vs. label model-13
MMLU-Pro $\rightarrow$ MATH500	0.800 (6.6)	+0.088 [+0.071,+0.106]*	+0.101 [+0.086,+0.115]*	+0.020 [+0.004,+0.037]*	+0.005 [-0.018,+0.028]
GPQA $\rightarrow$ MATH500	0.680 (5.7)	+0.010 [-0.001,+0.021] $\approx$	+0.018 [+0.007,+0.029]*	-0.043 [-0.059,-0.028]*	-0.127 [-0.156,-0.098]*
MATH500 $\rightarrow$ MMLU-Pro	0.638 (2.1)	+0.005 [-0.010,+0.020] $\approx$	+0.014 [+0.008,+0.020] $\approx$	-0.028 [-0.043,-0.014]*	-0.053 [-0.081,-0.023]*
GPQA $\rightarrow$ MMLU-Pro	0.675 (6.1)	+0.019 [+0.008,+0.031]*	-0.004 [-0.014,+0.005] $\approx$	-0.035 [-0.048,-0.022]*	-0.080 [-0.109,-0.051]*
MATH500 $\rightarrow$ GPQA	0.348 (2.1)	-0.001 [-0.017,+0.015] $\approx$	+0.006 [+0.001,+0.012] $\approx$	-0.021 [-0.035,-0.007]*	-0.069 [-0.100,-0.039]*
MMLU-Pro $\rightarrow$ GPQA	0.374 (6.8)	+0.000 [-0.015,+0.015] $\approx$	+0.001 [-0.010,+0.013] $\approx$	-0.017 [-0.032,-0.001]*	-0.097 [-0.123,-0.071]*

at 0.833 on MATH500 versus 0.568 for the static top-1 judge (`scripts/diag_first_judge.py`); further acquisitions then *lower* accuracy toward the full-ensemble value (Fig. 2, left). The same rule hurts slightly for GPQA sources (0.582 vs. 0.617 on MATH500). Under shift DAS is below the full ensembles in five of six pairs.

Ablations. Table 7 (and Fig. 3 in the appendix) isolates DAS’s terms at the $\delta=0.01$ point. The dependence penalty is the component with a clear effect: replacing the conditional standard deviation by the marginal one leaves test accuracy unchanged (0.836/0.377/0.681 vs. 0.839/0.384/0.686) but needs 5.7/7.9/8.4 instead of 2.1/5.8/6.7 calls to reach the same cal target, because without it the policy keeps buying judges whose evidence is predictable from those already held. Removing the information term costs accuracy on MATH500 (0.818 at 9.9 calls). Imputation and diversity-only values do not help.

Controlled redundancy: copied verifiers (synthetic stress test). To test the dependence penalty directly, we add $m \in \{1, 3\}$ artificial near-copies of the dev-best judge (its raw scores plus $\mathcal{N}(0, 0.01^2)$ noise, clipped; dev evidence correlation with the original > 0.99) to the real 13-judge pool, refit everything on dev, freeze $\delta=0.01$ thresholds on cal and evaluate on test over 10 resplits (Table 8; `scripts/stress_duplicates.py`). These copies are synthetic and labelled as such. With three copies, the redundant group (the original and its copies) is acquired 1.66/1.53/0.69 times per question by DAS, against 3.31/2.74/2.94 without the dependence term and 2.66/2.87/2.44 for confidence-only stopping. Test accuracy is unchanged or slightly higher (0.844/0.384/0.689 vs. 0.844/0.381/0.677). The penalty does what it is designed to do: it skips judges whose evidence is already held. Total calls are a noisier measure. Copies also change the cal curve from which the δ -rule threshold is chosen, and on MATH500 DAS’s mean calls rise by $+3.4 \pm 1.6$ with three copies (other datasets: $+1.5 \pm 1.4$, $+0.3 \pm 0.8$). Larger copy noise ($\sigma=0.05$) gives the same picture.

Candidate-level acquisition: a shortlist cascade (the falsifying experiment). If per-question adaptivity has value here, judge-over- N granularity may be what hides it. We therefore test acquisition at the candidate level with the simplest such policy, a two-stage shortlist cascade in the spirit of cascade ranking and retrieve-then-rerank [Wang et al., 2011, Nogueira and Cho, 2019]. The dev-best judge scores all $N=100$ candidates. Only its top- m candidates, a per-question and data-dependent set, are then scored by the next $k-1$ judges in dev order. A listwise model fit on dev shortlists selects among them. The cost is $100 + (k-1)m$ candidate-judge evaluations. The supervision matches label-assisted DAS (dev labels for the order and weights, cal labels for the operating point). For each resplit, the budget is DAS’s mean *cal* cost at its frozen operating point, and (m, k) with $m \in \{3, 5, 10, 20\}$, $k \leq 13$ is chosen on cal within that budget (`scripts/cascade_eval.py`). Table 9 gives the result. In-domain, the cascade beats DAS at the $\delta=0.01$ budget by $+0.014$ [$+0.000, +0.030$], $+0.045$ [$+0.021, +0.068$] and $+0.040$ [$+0.016, +0.064$], while spending 139/149/185 instead of 214/584/671 evaluations per question; the full ensemble spends 1300. It matches or beats the full 13-judge listwise ensemble ($-0.011/+0.022/+0.028$) at 86–89% fewer evaluations. It also ties Weaver’s label model on MMLU-Pro ($+0.001$ [$-0.024, +0.025$]), beats it on MATH500 ($+0.048$) and is below it on GPQA (-0.060). Under shift the cascade beats DAS on three pairs and ties on

Table 6: Cross-domain evaluation, per-resplit view (equal cost; $\delta=0.01$; mean over resplits; last column mean $\pm$ s.e. paired difference against cost-matched confidence-only).

Source→Target	n	Weaver-13	full-13 (listwise)	conf.-only (calls)	DAS (calls)	DAS+mean sel. (calls)	Δ DAS vs. conf.-only
MMLU-Pro→MATH500	500	0.748	0.780	0.684 (6.3)	0.800 (6.6)	0.792 (3.7)	+0.088 $\pm$ 0.032
GPQA→MATH500	500	0.768	0.724	0.681 (7.3)	0.680 (5.7)	0.613 (1.5)	+0.010 $\pm$ 0.013
MATH500→MMLU-Pro	500	0.672	0.666	0.638 (3.3)	0.638 (2.1)	0.652 (1.0)	+0.005 $\pm$ 0.011
GPQA→MMLU-Pro	500	0.681	0.710	0.673 (7.5)	0.675 (6.1)	0.601 (1.6)	+0.019 $\pm$ 0.020
MATH500→GPQA	646	0.467	0.373	0.355 (2.8)	0.351 (2.1)	0.365 (1.0)	-0.001 $\pm$ 0.004
MMLU-Pro→GPQA	646	0.467	0.393	0.377 (6.3)	0.376 (6.8)	0.368 (3.8)	+0.001 $\pm$ 0.007

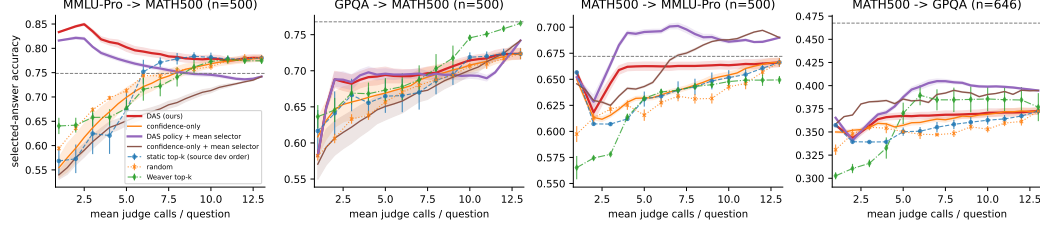


Figure 2: Cross-domain accuracy versus calls (four of six pairs). Dashed grey: upstream-protocol Weaver with all 13 judges on the target.

two. On MMLU-Pro→MATH500 it loses (-0.099), because its first judge is the static top-1 judge, the very weakness that DAS’s first-judge rule avoided above. The controls in Table 10 locate the gain; that table fixes $m=10$ and uses all judges, a configuration that was picked after a look at one test split and is used only for these relative comparisons, never as a headline. Removing the shortlist ($m=N$) reproduces the full listwise ensemble exactly (implementation check). A random shortlist of the same size is far worse (0.789/0.367/0.624 vs. 0.876/0.458/0.737). Permuting dev labels within shortlists removes the gain (0.831/0.375/0.661). The gain therefore comes from letting one strong judge discard most distractor answer clusters before the other judges are consulted, not from leakage or from shortlisting as such. The shortlist keeps a correct candidate for 0.950/0.729/0.839 of questions, which bounds what the cascade can reach.

Calibration and cost model. The Platt-mapped stopping statistic is reasonably calibrated in-domain but badly miscalibrated under shift (ECE up to 0.518 for MATH500→GPQA; appendix). Under the declared parameter-count cost model (three resplits) DAS shifts spend toward the 2–3B judges, spending 150/568/364 units per question versus 1300 for the equal-cost full ensemble, at 0.838/0.397/0.662 accuracy.

7 Discussion

The study was designed so that per-question adaptivity could win: correlated judges, an unweighted consensus that degrades with more judges, large oracle headroom, and a policy that sees only masked scores. At judge-over- N granularity it did not. Almost all of the available saving is captured by choosing a small judge subset *once* on development data, and a per-question stopping rule then adds noise to the operating point rather than accuracy. Two mechanisms limit judge-level adaptivity. The first acquisition is necessarily question-independent, and for MATH500 one judge already carries most of the signal. The stopping statistic is computed from the same few judges whose errors it is supposed to detect, and under shift it is miscalibrated. What does help is spending the remaining judges where the decision is actually made. After one judge, most of the 100 candidates are no longer in contention, and scoring only the shortlist both cuts cost by roughly an order of magnitude and *raises* accuracy, because a reranker trained on shortlists is not distracted by large wrong-answer clusters. This is a candidate-level form of adaptive verification. It is the regime the original motivation pointed to, but it reaches it through which candidates are scored rather than which judges are consulted. Combining the cascade with DAS’s first-judge rule (which fixes the cascade’s one shift failure) and with the label-model selector (strongest on GPQA) are the obvious next steps.

Table 7: Ablations, in-domain test, $\delta=0.01$ operating point (mean $\pm$ s.e. over resplits).

Variant	MATH500		GPQA		MMLU-Pro	
	acc	calls	acc	calls	acc	calls
DAS (full)	0.839 $\pm$ 0.006	2.1	0.384 $\pm$ 0.011	5.8	0.686 $\pm$ 0.005	6.7
w/o dependence penalty (marginal sd)	0.836 $\pm$ 0.007	5.7	0.377 $\pm$ 0.010	7.9	0.681 $\pm$ 0.011	8.4
w/o reliability/partial-information term	0.818 $\pm$ 0.010	9.9	0.386 $\pm$ 0.008	8.8	0.677 $\pm$ 0.011	9.7
diversity-only value	0.828 $\pm$ 0.007	6.2	0.388 $\pm$ 0.012	4.1	0.680 $\pm$ 0.007	5.9
+ conditional-mean imputation	0.837 $\pm$ 0.005	2.1	0.384 $\pm$ 0.008	5.8	0.685 $\pm$ 0.006	8.1
DAS policy, unweighted-mean selector	0.814 $\pm$ 0.005	1.0	0.376 $\pm$ 0.006	1.5	0.680 $\pm$ 0.005	3.7
label-free policy (score covariance), mean selector	0.776 $\pm$ 0.011	2.4	0.370 $\pm$ 0.009	2.1	0.662 $\pm$ 0.010	4.5
WS-weighted continuous selector (unlabeled)	0.646 $\pm$ 0.010	2.9	0.350 $\pm$ 0.008	2.5	0.750 $\pm$ 0.009	1.9
label-model (binarised) selector, unlabeled weights	0.799 $\pm$ 0.014	4.2	0.448 $\pm$ 0.008	7.8	0.700 $\pm$ 0.015	10.2
confidence-only (fixed dev order)	0.839 $\pm$ 0.005	4.2	0.385 $\pm$ 0.009	7.1	0.675 $\pm$ 0.012	6.2
confidence-only, mean selector	0.814 $\pm$ 0.005	1.2	0.371 $\pm$ 0.010	4.4	0.672 $\pm$ 0.011	4.4
confidence-only, label-model selector	0.793 $\pm$ 0.017	6.3	0.450 $\pm$ 0.010	10.2	0.708 $\pm$ 0.013	11.5

Table 8: Synthetic copied-verifier stress test (mean over 10 resplits; $\delta=0.01$ operating points frozen on cal). “group”: members of the redundant group (original + copies) acquired per question before stopping.

Dataset (copied judge)	copies	DAS: acc / calls / group	DAS w/o dep.: acc / calls / group	conf.-only: acc / calls / group
MATH500 (QwenPRM)	0	0.839 / 2.1 / –	0.836 / 5.7 / –	0.839 / 4.2 / –
MATH500 (QwenPRM)	1	0.843 / 4.1 / 1.18	0.837 / 5.2 / 1.69	0.839 / 5.4 / 1.60
MATH500 (QwenPRM)	3	0.844 / 5.6 / 1.66	0.844 / 7.6 / 3.31	0.837 / 6.8 / 2.66
MATH500 (QwenPRM)	3 ($\sigma=0.05$)	0.845 / 4.6 / 1.70	0.839 / 7.4 / 3.43	0.838 / 6.7 / 2.66
GPQA (ArmorRM/EurusPRMStage1/InternLM2Reward7B/QRM)	0	0.384 / 5.8 / –	0.377 / 7.9 / –	0.385 / 7.1 / –
GPQA (ArmorRM/EurusPRMStage1/InternLM2Reward7B/QRM)	1	0.384 / 6.6 / 0.92	0.380 / 9.9 / 1.57	0.382 / 9.3 / 1.73
GPQA (ArmorRM/EurusPRMStage1/InternLM2Reward7B/QRM)	3	0.384 / 7.3 / 1.53	0.381 / 10.4 / 2.74	0.380 / 9.1 / 2.87
GPQA (ArmorRM/EurusPRMStage1/InternLM2Reward7B/QRM)	3 ($\sigma=0.05$)	0.389 / 6.8 / 1.57	0.380 / 9.6 / 2.59	0.368 / 6.3 / 2.06
MMLU-Pro (ArmorRM/GRM/GRMLlama32)	0	0.686 / 6.7 / –	0.681 / 8.4 / –	0.675 / 6.2 / –
MMLU-Pro (ArmorRM/GRM/GRMLlama32)	1	0.691 / 7.9 / 0.76	0.679 / 8.5 / 1.52	0.675 / 7.8 / 1.70
MMLU-Pro (ArmorRM/GRM/GRMLlama32)	3	0.689 / 7.0 / 0.69	0.677 / 9.4 / 2.94	0.675 / 7.2 / 2.44
MMLU-Pro (ArmorRM/GRM/GRMLlama32)	3 ($\sigma=0.05$)	0.683 / 6.4 / 0.75	0.676 / 9.4 / 3.01	0.674 / 7.3 / 2.00

8 Limitations

Cached scores, call counts only. All verifier outputs are cached; we report calls per question (one call = one judge over N candidates = N candidate–judge evaluations) and a declared parameter-count cost model. We measured no latency, dollars or energy, and we did not validate the declared cost model against live inference. **Granularity and cost model.** DAS acquires judge-over- N ; the candidate-level cascade assumes that scoring m candidates costs m/N of a full judge call, which holds for per-candidate reward models but ignores batching and latency effects we could not measure on cached scores. The cascade’s operating point is chosen from 52 (m, k) configurations on 100 cal questions, and its first judge is the static dev top-1, which fails under one shift. **Distilled verifier.** We evaluated the released distilled checkpoints, but they were trained on 70B generations; we did not re-distil a cross-encoder on 8B pseudo-labels, which would be the fair efficiency comparison (estimated a few GPU-hours). **Statistical power.** In-domain test sets have 200–258 questions per resplit; ten resplits and a question-level pooled analysis resolve differences of about 1.5 points, and the equivalence margin of 2 points is a declared, not a derived, tolerance. Seeds are question-level resplits, so seed-to-seed spread reflects split variance, not training randomness; the cross-domain pooled CIs resample target questions only. **Labels in the pipeline.** The label-assisted variants use dev labels for reliabilities and weights; every variant, including the “label-free” policy, uses cal labels for the stopping threshold and the class-balance scalar, as upstream Weaver does for its binarisation threshold and class balance; a deployment without any labelled questions would have to set ε by budget. **Protocol differences from upstream.** Upstream Weaver is transductive (fit on the evaluation set) and uses dataset-specific thresholds tuned on those sets; our Weaver numbers on our splits use cal-selected thresholds and cal class balance, and we report the upstream-threshold variant separately. **Surrogate assumptions.** The Gaussian evidence surrogate ignores within-question dependence between candidates and uses a myopic one-step value; the flip probability is a heuristic, not a calibrated posterior, as the ECE values show. **Generator and verifier pool.** One generator (Llama-3.1-8B-Instruct), 13 open reward/process-reward models, no LM judges; results on the 70B tables with 33 verifiers may differ.

Table 9: Shortlist cascade (candidate-level acquisition) vs. DAS at DAS’s frozen $\delta=0.01$ cal budget, question-level pooled over 10 resplits (95% CI; * significant, $\approx$ equivalent within ± 0.02). “evals”: candidate–judge evaluations per question (full ensemble: 1300).

Eval	cascade acc (evals)	DAS acc (evals)	vs. DAS	vs. static top- k	vs. full-13 listwise	vs. label model-13
MATH500 test	0.853 (139)	0.839 (214)	+0.014 [+0.000,+0.030]*	+0.021 [+0.007,+0.035]*	-0.011 [-0.031,+0.010]	+0.048 [+0.021,+0.075]*
GPQA test	0.434 (149)	0.389 (584)	+0.045 [+0.021,+0.068]*	+0.069 [+0.048,+0.090]*	+0.022 [-0.003,+0.048]	-0.060 [-0.091,-0.031]*
MMLU-Pro test	0.720 (185)	0.680 (671)	+0.040 [+0.016,+0.064]*	+0.057 [+0.037,+0.077]*	+0.028 [+0.004,+0.052]*	+0.001 [-0.024,+0.025]
MMLU-Pro→MATH500	0.702 (186)	0.800 (659)	-0.099 [-0.121,-0.076]*	+0.128 [+0.108,+0.150]*	-0.078 [-0.101,-0.056]*	-0.093 [-0.117,-0.071]*
GPQA→MATH500	0.686 (150)	0.680 (573)	+0.005 [-0.012,+0.023]	+0.051 [+0.035,+0.068]*	-0.038 [-0.059,-0.017]*	-0.122 [-0.148,-0.094]*
MATH500→MMLU-Pro	0.673 (139)	0.638 (213)	+0.035 [+0.020,+0.051]*	+0.031 [+0.017,+0.047]*	+0.007 [-0.011,+0.025]	-0.018 [-0.042,+0.007]
GPQA→MMLU-Pro	0.672 (150)	0.675 (610)	-0.003 [-0.021,+0.016] $\approx$	+0.052 [+0.037,+0.066]*	-0.037 [-0.059,-0.016]*	-0.083 [-0.110,-0.056]*
MATH500→GPQA	0.375 (139)	0.348 (211)	+0.026 [+0.010,+0.042]*	+0.025 [+0.010,+0.041]*	+0.005 [-0.014,+0.024]	-0.043 [-0.072,-0.014]*
MMLU-Pro→GPQA	0.422 (186)	0.374 (682)	+0.047 [+0.025,+0.070]*	+0.059 [+0.042,+0.075]*	+0.030 [+0.008,+0.053]*	-0.050 [-0.074,-0.025]*

Table 10: Cascade mechanism controls (in-domain test, $m=10$, all 13 judges on the shortlist; mean $\pm$ s.e. over 10 resplits). Not used for headline numbers.

Dataset	first-judge shortlist	no shortlist ($m=N$)	random shortlist	shuffled dev labels	shortlist recall
MATH500	0.876 $\pm$ 0.006	0.859 $\pm$ 0.006	0.789 $\pm$ 0.005	0.831 $\pm$ 0.008	0.950 $\pm$ 0.004
GPQA	0.458 $\pm$ 0.018	0.409 $\pm$ 0.007	0.367 $\pm$ 0.005	0.375 $\pm$ 0.017	0.729 $\pm$ 0.012
MMLU-Pro	0.737 $\pm$ 0.011	0.698 $\pm$ 0.005	0.624 $\pm$ 0.012	0.661 $\pm$ 0.009	0.839 $\pm$ 0.008

Reproducibility statement

All code is in `src/p04` (adapter, environment, selectors, dependence model, policies, Weaver reimplementations), with the exact upstream pins, HF revisions and SHA-256 hashes in `sources.lock.json`. Every number in this paper is read from `results/aggregate/*/summary.json`, produced by `scripts/aggregate.py` from immutable per-question records under `results/p04/*`, and typeset by `scripts/make_tables.py`; every figure is produced by `scripts/make_figures.py`. `README.md` lists the commands (setup, smoke, baseline, candidate, evaluate, report); `EXPERIMENTS.md` is the append-only log including the failed aggregator variants. All selection experiments ran on a laptop CPU with two threads; the distilled-verifier scoring ran on Modal L4 GPUs (`modal_jobs/distilled_verifier.py`; cost in `cost_ledger.csv`).

References

- Stephen H. Bach, Bryan He, Alexander Ratner, and Christopher Ré. Learning the structure of generative models without labeled data. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70, pages 273–282, 2017.
- 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.
- Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.
- Daniel Y. Fu, Mayee F. Chen, Frederic Sala, Sarah M. Hooper, Kayvon Fatahalian, and Christopher Ré. Fast and three-rious: Speeding up weak supervision with triplet methods. *ICML*, 2020.
- Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *Advances in Neural Information Processing Systems*, 30, 2017.
- Erfan Hajishemi and Yanning Shen. Online verification of language model responses under cost constraints. *arXiv preprint arXiv:2610.02632*, 2026.
- Ronald A. Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 2(1):22–26, 1966.
- Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, 9:235–284, 2008.

Olivier Ledoit and Michael Wolf. A well-conditioned estimator for large-dimensional covariance matrices. *Journal of Multivariate Analysis*, 88(2):365–411, 2004.

Shalev Lifshitz, Sheila A. McIlraith, and Yilun Du. Multi-agent verification: Scaling test-time compute with multiple verifiers. In *arXiv preprint arXiv:2502.20379*, 2025.

Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*, 2019.

Alexander Ratner, Christopher De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. *Advances in Neural Information Processing Systems*, 29, 2016.

Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Ré. Training complex models with multi-task weak supervision. In *AAAI*, 2019.

Jon Saad-Falcon, E. Kelly Buchanan, Mayee F. Chen, Tzu-Heng Huang, Brendan McLaughlin, Tanvir Bhathal, Shang Zhu, Ben Athiwaratkun, Frederic Sala, Scott Linderman, Azalia Mirhoseini, and Christopher Ré. Shrinking the generation-verification gap with weak verifiers. *arXiv preprint arXiv:2506.18203*, 2025.

Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314*, 2024.

Neeraj Varshney and Chitta Baral. Model cascading: Towards jointly improving efficiency and accuracy of nlp systems. *EMNLP*, 2022.

Lidan Wang, Jimmy Lin, and Donald Metzler. A cascade ranking model for efficient ranked retrieval. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 105–114, 2011.

A Additional tables

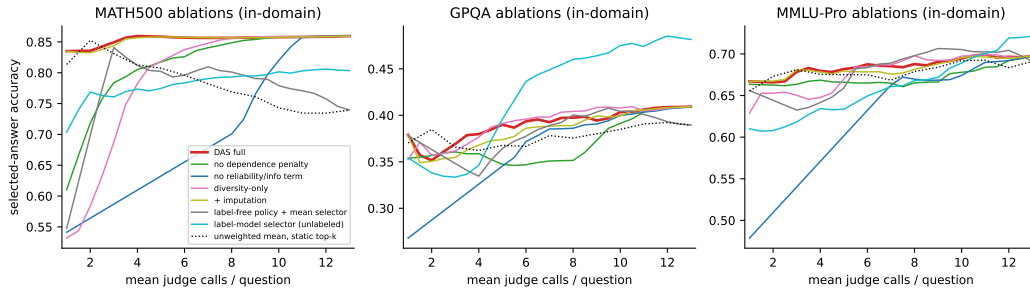


Figure 3: Ablation curves (in-domain test, mean over resplits).

Calibration and risk-coverage.

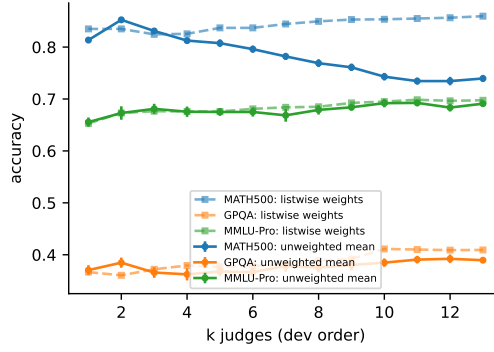


Figure 4: Static top- k in dev order: unweighted mean (solid) versus listwise weights (dashed). On MATH500 adding judges to an unweighted average lowers accuracy.

Eval	Policy	acc	ECE	AURC	calls
MATH500 test	das sup	0.839	0.033	0.181	2.1
MATH500 test	conf only sup	0.839	0.056	0.063	4.2
MATH500 test	das wsv	0.799	0.056	0.177	4.2
GPQA test	das sup	0.384	0.052	0.605	5.8
GPQA test	conf only sup	0.385	0.061	0.602	7.1
GPQA test	das wsv	0.448	0.054	0.514	7.8
MMLU-Pro test	das sup	0.686	0.035	0.293	6.7
MMLU-Pro test	conf only sup	0.675	0.076	0.313	6.2
MMLU-Pro test	das wsv	0.700	0.038	0.297	10.2
MMLU-Pro→MATH500	das sup	0.800	0.109	0.197	6.6
MMLU-Pro→MATH500	conf only sup	0.684	0.097	0.304	6.3
MATH500→GPQA	das sup	0.351	0.518	0.639	2.1
MATH500→GPQA	conf only sup	0.355	0.502	0.641	2.8

Cost-model sensitivity. The declared *params* model prices a judge by its parameter count relative to 8B (GRM-Gemma 2B = 0.25, GRM-Llama-3.2 3B = 0.375, 7-8B $\approx$ 0.9-1); DAS divides value by cost, so cheaper judges are preferred when their information is comparable.

Cost model	Dataset	DAS acc	DAS cost/q (units)	full-13 acc	full-13 cost/q
equal	MATH500	0.839 $\pm$ 0.006	214	0.859	1300
equal	GPQA	0.384 $\pm$ 0.011	581	0.409	1300
equal	MMLU-Pro	0.686 $\pm$ 0.005	667	0.698	1300
params (declared)	MATH500	0.838 $\pm$ 0.013	150	0.867	—
params (declared)	GPQA	0.397 $\pm$ 0.008	568	0.419	—
params (declared)	MMLU-Pro	0.662 $\pm$ 0.020	364	0.703	—