ON THE ROBUSTNESS OF TARGETED INSTRUCTION DATA SELECTION AND BEYOND

Anonymous authors
Paper under double-blind review

ABSTRACT

Targeted instruction data selection and target-agnostic instruction selection are treated as two separate lines of work, yet they meet at a natural limit: when the target set is entirely absent or carries no task signal, a targeted selector reduces to a target-agnostic one. In practice, this limit is approached more often than it appears. Target sets collected from real deployments are typically small or may contain redundant or mislabeled examples, causing the estimated target profile to be noisy and unstable. A matching signal built on such a profile can steer selection away from high-quality source data and toward noise, producing worse results than simply ignoring the target set. We call a targeted selector *robust* if it consistently beats target-agnostic selection across varying target sizes and noise levels. Measured by this criterion, most existing targeted selectors are not robust: four of the five we evaluate score below a target-agnostic baseline, meaning the target set hurts rather than helps. This failure is largely invisible in existing benchmarks, because targeted methods are evaluated against random selection rather than against target-agnostic selection. We identify two structural causes and address them with our method, CLASS (Closed-set LAbel Space Selection), which combines a quality term to supply the target-agnostic floor with closed-set capability matching to keep profile estimation stable under both target scarcity and target noise across different benchmarks. CLASS reaches an average of 54.1 across five tasks, above the target-agnostic baseline at 52.0 and above every targeted baseline, with highest selection stability.

1 INTRODUCTION

Instruction tuning adapts large language models to downstream use, and the composition of the finetuning mixture often matters more than its size (Wang et al., 2023b; Wei et al., 2021; Zhou et al., 2023; Chen et al., 2023a). This has produced a large literature on instruction data selection, which splits into two branches that rarely meet.

Target-agnostic selection scores each source sample by intrinsic properties such as writing quality, complexity, or difficulty, usually with an LLM judge or a small proxy model, and keeps the top-ranked samples (Chen et al., 2023a; Liu et al., 2023; Li et al., 2024c;b; Zhao et al., 2024). It needs no knowledge of the downstream task and produces one general-purpose subset. To avoid ambiguity, we use the term *quality-only selection* throughout this paper to refer specifically to target-agnostic methods that employ an LLM judge to score source samples.

Targeted selection instead assumes a small set of target examples that characterizes the task, and ranks source samples by similarity to that set: by low-rank gradient alignment (Xia et al., 2024), by hidden-state embedding similarity (Ivison et al., 2025), by optimal transport between source and target samples (Liu et al., 2024b), or by matching over LLM-generated proxy labels (Cheng et al., 2026). Most methods in this branch are, at bottom, matching rules.

The two branches are usually compared only at their own preferred operating points, and the target set is treated as a fixed, reliable description of the task. In deployment it is neither: a practitioner adapting a model to a new task typically has a handful of examples rather than a curated validation split, and those examples may be unrepresentative, redundant, or mislabeled.

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.

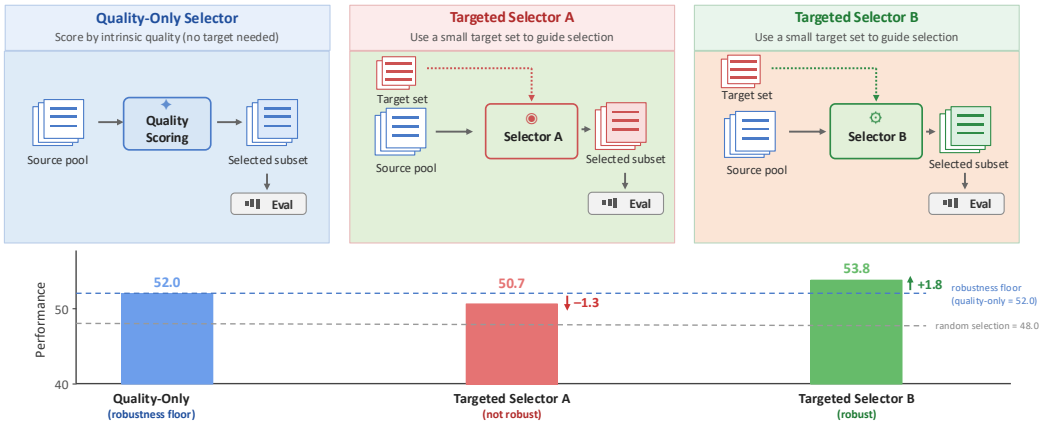


Figure 1: **Robustness connects target-agnostic and targeted selection.** Quality-only selection (*left*) scores source samples by intrinsic quality with no access to a target task. Targeted selectors A and B (*center, right*) additionally draw a small target set to steer source matching. All three are evaluated on the same benchmark suite; the quality-only score defines the *robustness floor* (dashed line). Selector A outperforms random selection but falls below quality-only, meaning the target set actively hurts relative to ignoring it. Selector B clears the floor: merely outperforming random selection is not enough; a robust targeted selector should beat the target-agnostic baseline.

This paper connects the two branches through a robustness argument. When target examples carry no task signal, a targeted selector can do no better than a target-agnostic one. In the limit where the target set is empty or every example is noise, the problem reduces to target-agnostic selection: the two branches are two ends of one axis, parameterized by how much task information the target contains. A targeted method is *robust* when it beats the target-agnostic selector across this axis. Concretely, the right baseline is not random selection but quality-only selection: the question is not whether the target set helps relative to doing nothing, but whether it helps relative to the method that ignores it entirely (Figure 1).

Measured by this criterion, most existing targeted selectors are not robust. Section 5 shows that at 50 target examples, four of the five targeted baselines we study score below a quality-only baseline that never looks at the target. This failure is nearly invisible in existing benchmarks, where targeted methods are compared to random selection rather than to quality-only selection. Prior work has noticed pieces of this picture: (Pang et al., 2024) find that a target-agnostic quality selector outperforms targeted selection in their setting, yet this observation has not been connected to a broader principle or used to reframe how targeted methods are evaluated.

Two structural reasons account for most of the deficit. First, most existing targeted rules are pure matching rules with no target-independent component, so they have no defined floor when the target carries no information. Second, their matching signals live in high-dimensional continuous spaces where a target estimate built from few examples is unstable, reordering many source candidates relative to the ground-truth ranking. Adding a quality score on top of an existing matching signal is not enough: when the matching term is unstable it can pull selection away from the quality floor, and a further practical cost is that stacking a quality-scoring LLM on top of a per-sample matching pipeline requires two separate inference passes over the source pool. A robust solution requires a fundamentally different matching space, one where few examples suffice to estimate the target reliably and where quality and matching can be obtained together in a single pass.

We propose matching in a closed-set label space. A fixed vocabulary of K capability labels reduces the target to a K-dimensional aggregate profile and matching is computed against the aggregate rather than against individual target examples, per-sample noise in the target set largely cancels in the mean, making the signal far more robust to both target scarcity and noise than per-sample methods. Crucially, the quality score and the capability profile are produced by the same single LLM call per

source sample and reused across all targets, so CLASS adds less inference overhead beyond the cost of quality-only selection. Our contributions are:

- We are the first to introduce a robustness view of targeted instruction data selection by framing quality-only selection as the floor any targeted method should exceed, and show that most existing methods fall below it under both target scarcity and target noise.
- We propose CLASS, an efficient selection rule where a single LLM call per source sample produces both a quality score and a capability profile, and show that CLASS is the only method in our study that beats quality-only on every benchmark under both reduced target size and two types of target noise.

2 Related Work

Target-agnostic instruction selection. This branch scores instruction samples without reference to any downstream task. AlpaGasus rates each sample with an LLM judge (Chen et al., 2023a), DEITA scores complexity and quality jointly with a diversity filter (Liu et al., 2023), IFD measures how much an instruction reduces the difficulty of producing its response (Li et al., 2024c) and Superfiltering computes it with a small proxy model (Li et al., 2024b), LIMA shows the value of a small curated set (Zhou et al., 2023), and response length alone is a strong baseline (Zhao et al., 2024). Others use LLM-free proxies (Cao et al., 2023; He et al., 2024) or broader LLM-driven curation (Pang et al., 2024; Liu et al., 2024a). None depends on a target set, which is why we use one as the reference point of the axis rather than as a competing method. Among these methods, LLM-as-judge approaches consistently achieve the strongest results.

Targeted instruction selection. This second branch uses target examples to steer selection, and each method is defined by the space in which it matches: projected per-sample gradients aligned with the target gradient in LESS (Xia et al., 2024; Koh & Liang, 2017; Ilyas et al., 2022; Engstrom et al., 2024), cosine similarity of weighted-mean-pooled hidden states in RDS+ (Ivison et al., 2025), optimal transport between individual source and target samples with a diversity regularizer in TSDS (Liu et al., 2024b; Cuturi, 2013), and proxy tags clustered on the target and propagated onto the source pool in TADS (Cheng et al., 2026). Coreset and importance-reweighting formulations share the per-sample matching structure (Mirzasoleiman et al., 2020; Killamsetty et al., 2021; Yao et al., 2022; Xie et al., 2023). All are pure matching rules, so none has a defined behaviour when the target carries no information, and none treats quality as part of its objective; quality appears at most as a prior filter (Cheng et al., 2026). (Nayak et al., 2026) ask how much of the reported benefit comes from the matching algorithm rather than the representation; our study is complementary, varying the amount of target information while holding the quality signal fixed across mechanisms.

Label structures and few-sample estimation. Intermediate label structures have been used to describe and steer instruction data, as an analysis tool over thousands of open-vocabulary tags (Lu et al., 2023), to study how skill mixtures affect learning (Chen et al., 2023b; Didolkar et al., 2024), to drive data generation from a capability taxonomy (Li et al., 2024a), and to organize post-training data around a few core skills (Lambert et al., 2024). These treat the label space as a descriptive or generative device. We use it instead as the matching space in which few target examples suffice to estimate the aggregate target profile, drawing on the same intuition that motivates class-prior correction under label shift (Zhang et al., 2013; Menon et al., 2020; Wallace et al., 2011): a low-dimensional categorical distribution is far easier to estimate from few samples than a high-dimensional continuous one.

3 The Robustness Axis

3.1 Setup

Let $D = \{x_i\}_{i=1}^{N}$ be a source pool of instruction-response pairs and E a held-out evaluation set for a downstream task. A selector receives a target set T with $m = |T|$ examples from the task and returns $S \subseteq D$ with $|S| = n$, which finetunes a base model under a fixed recipe. Write $S_A(T)$ for the subset selector A returns and $U(S)$ for its evaluation score on E after model finetuning. The *quality-only*

selector S_Q ignores T entirely and solves

$$S_Q \;=\; \underset{S\subseteq D,\,|S|=n}{\arg\max}\; \frac{1}{|S|}\sum_{x\in S} s(x), \tag{1}$$

where $s(x)$ is an intrinsic quality score, serving as the target-agnostic reference. Throughout, N is the source size, m the target size, n the budget; E is never used to build T or tune hyperparameters.

The primary axis of this paper is the amount of task information the target set carries, varied through two routes: reducing target size m and injecting target noise. This is deliberately not the same as shrinking n: reducing n removes training data, so a drop in U is expected, while varying the target set changes only the information used to decide which data to train on, and every model still sees n examples. Both routes converge toward the same limit where the target carries no task signal and the problem collapses to target-agnostic selection.

3.2 What to Measure

We measure downstream utility via the targeting gain

$$\mathrm{Gain}_A(m) \;=\; U\big(S_A(T_m)\big) \;-\; U(S_Q), \tag{2}$$

the improvement over discarding the target entirely. A negative value means the practitioner would have done better with a method that never looks at the target. We call a selector *robust* if it satisfies $\mathrm{Gain}_A(m) > 0$ across benchmark and across the target scarcity and target noise, meaning it consistently outperforms the target-agnostic baseline. We do not require this to hold at the extremes of $m = 0$ or fully corrupted targets, as these limits are outside the range of practical deployment: a practitioner who has no usable target examples should use a target-agnostic method directly. We therefore measure robustness at practical settings (Section 5.1). Selection stability, which measures how consistently a selector picks the same data across different target draws, is reported as a secondary diagnostic in Section 5.4.

3.3 Adding Quality to Per-Sample Matching

To make a per-sample matching selector robust, a natural fix is to directly add a quality score to the matching signal:

$$\mathrm{Score}_A(x;T,S) \;=\; \lambda\, s(x) \;+\; \alpha\, b_A(x;T,S), \tag{3}$$

where s is the quality score, b_A is a matching signal, λ and α are hyper-parameters, and $\alpha = 0$ recovers Eq. (1) exactly.

In practice, when the target size is small or target examples are noisy or out of distribution, the matching signal becomes less representative, and the selection drifts away from the quality floor. Two features make a matching signal fragile under either condition: if b_A depends on individual target examples rather than an aggregate, removing examples or injecting noise changes it directly, and if the aggregate lives in a high-dimensional continuous space, its few-sample estimate is poorly conditioned relative to the spread of scores across candidates, so a small error reorders many of them. Gradient and embedding matching have both features whereas a fixed low-dimensional label space exhibits neither. This motivates CLASS, presented in the next section.

4 CLASS: A Selection Rule Built for Robustness

4.1 Capability Profiles in a Closed-Set Label Space

We define a fixed label space

$$\mathcal{Z} = \{z_1, z_2, \ldots, z_K\},$$

where each z_k is a predefined capability label. Unlike proxy labels obtained through clustering or propagation (Lu et al., 2023; Cheng et al., 2026), $\mathcal{Z}$ is chosen before any data is seen and remains identical across all source pools and target tasks; this stability keeps source- and target-side distributions directly comparable and makes source-side annotation reusable with no per-target recomputation.

By default $K = 12$, with the labels Reasoning, Knowledge, Extraction, Summarization, Rewriting, Classification, Translation, Coding, Planning, Dialogue, Safety, and Creative. This granularity sits between the seven core skills of TULU 3 (Lambert et al., 2024) and the thousands of open-vocabulary tags of InsTag (Lu et al., 2023); Section 5.5 varies $K \in \{3, 12, 47\}$.

For each sample x, a single LLM call (Zheng et al., 2023) returns a relevance score $a_z(x) \in \{0, \dots, 10\}$ for every $z \in \mathcal{Z}$ and, for source samples, an overall quality score $s(x) \in \{1, \dots, 10\}$ playing the role of s in Eq. (1). Both signals come from one prompt (Appendix C.5), so CLASS adds less inference overhead beyond quality-only selection. The capability profile is the soft distribution

$$q(z \mid x) \;=\; \frac{\exp(a_z(x))}{\sum_{z' \in \mathcal{Z}} \exp(a_{z'}(x))}, \qquad z \in \mathcal{Z}, \tag{4}$$

which captures the fact that instruction samples routinely exercise several capabilities at once. Source-side labeling is target-agnostic and done once; switching to a new target requires only re-labeling the small target set, with no new source-side LLM inference.

Coverage. Some samples are not described well by any label in $\mathcal{Z}$. Rather than a catch-all category, which would absorb heterogeneous content into one coordinate, we mark them uncovered through $u(x) = \mathbf{1}\big[\max_z a_z(x) < \tau\big]$ with $\tau = 5$: they are excluded from profile estimation but stay in the candidate pool, where the quality term can still select them. Write D_{cov}, S_{cov}, and T_{cov} for the covered parts of the pool, the selection, and the target.

4.2 DISTRIBUTION MATCHING OBJECTIVE

Let $S_{\text{cov}} = \{x \in S : u(x) = 0\}$ and $T_{\text{cov}} = \{x \in T : u(x) = 0\}$ denote the covered subsets of the selection and target. We define the label distributions

$$\pi_S(z) \;=\; \frac{1}{|S_{\text{cov}}|} \sum_{x \in S_{\text{cov}}} q(z \mid x), \qquad \pi_T(z) \;=\; \frac{1}{|T_{\text{cov}}|} \sum_{x \in T_{\text{cov}}} q(z \mid x), \tag{5}$$

and the matching loss

$$\mathcal{L}_{\text{match}}(S) \;=\; \tfrac{1}{2} \left\| \pi_S - \pi_T \right\|_2^2. \tag{6}$$

Minimizing $\mathcal{L}_{\text{match}}$ encourages the selected subset to have the same label composition as the target. Crucially, T enters the objective only through its aggregated profile π_T, so per-sample noise in T is averaged out before it influences selection, and the influence of any one target example is $O(1/m)$.

A second goal is high quality. We encode this through the linear quality functional

$$\mathcal{L}_{\text{qual}}(S) \;=\; -\frac{1}{|S|} \sum_{x \in S} s(x), \tag{7}$$

whose minimization promotes samples with higher LLM-assigned quality. Combining the two yields the overall objective

$$\min_{S \subseteq D,\, |S| = n} \; \alpha\, \mathcal{L}_{\text{match}}(S) \;+\; \lambda\, \mathcal{L}_{\text{qual}}(S), \tag{8}$$

with $\alpha, \lambda > 0$. We add no explicit diversity term: because π_S is an average over selection, a label that is already over-represented relative to π_T receives a negative deficit and is automatically discouraged.

4.3 GREEDY RULE AND ITS FIRST-ORDER JUSTIFICATION

Optimizing Eq. (8) exactly is combinatorial, so we build S greedily. Define the *label deficit* of the current selection

$$r_S(z) \;=\; \pi_T(z) - \pi_S(z), \tag{9}$$

the signed amount by which each capability is under-represented relative to the target. A direct calculation (Proposition 1) shows that adding a covered candidate x to S reduces $\mathcal{L}_{\text{match}}$ by

$$\frac{1}{|S_{\text{cov}}| + 1} \sum_{z \in \mathcal{Z}} q(z \mid x)\, r_S(z) \;+\; (x\text{-independent terms}) \;+\; O\big(1/|S_{\text{cov}}|^2\big),$$

so the candidate that maximizes $\langle q(\cdot \mid x),\, r_S \rangle$ also maximally reduces $\mathcal{L}_{\text{match}}$ at first order. Combining this with the quality bias gives the CLASS scoring rule

$$\text{Score}(x; S) \;=\; \lambda\, s(x) \;+\; \alpha\, \mathbf{1}[u(x) = 0]\, \big\langle q(\cdot \mid x),\, r_S \big\rangle. \tag{10}$$

This is exactly the additive form of Eq. (3), with the closed-set inner product $\langle q(\cdot \mid x),\, r_S \rangle$ instantiating the matching signal $b_A(x; T, S)$. For uncovered candidates the matching term vanishes and selection is driven purely by quality. The full procedure is stated as Algorithm 1 in Appendix C.1.

Proposition 1 (Exact reduction of the matching loss)**.** *Let S be a selection whose covered part has size $c = |S_{\text{cov}}| \geq 1$, and let $x \in D_{\text{cov}} \setminus S$. Then*

$$\mathcal{L}_{\text{match}}(S) - \mathcal{L}_{\text{match}}(S \cup \{x\}) = \frac{1}{c+1} \big\langle q(\cdot \mid x) - \pi_S,\, r_S \big\rangle \;-\; \underbrace{\frac{1}{2(c+1)^2} \big\| q(\cdot \mid x) - \pi_S \big\|_2^2}_{O(1/c^2)}. \tag{11}$$

Since $\langle \pi_S, r_S \rangle$ does not depend on x, the only x-dependent term of order $1/c$ is $\langle q(\cdot \mid x), r_S \rangle/(c+1)$, so maximizing it over candidates maximizes the reduction of $\mathcal{L}_{\text{match}}$ up to $O(1/c^2)$. For an uncovered candidate, S_{cov} is unchanged and the reduction is exactly zero.

The proof is in Appendix B.1. Eq. (10) is therefore greedy first-order descent on Eq. (8) with a quality bonus: the matching term selects candidates that fill the largest gaps in the current label profile, while the quality term ensures those gaps are filled by high-quality samples rather than by low-quality ones that happen to resemble the target.

4.4 Why CLASS Is Robust to Target Scarcity and Noise

Both failure modes identified in Section 3.3 are addressed by the same structural choice: T enters Eq. (10) only through the aggregate profile π_T, never through individual target examples. Under target scarcity, π_T is a mean over bounded K-dimensional vectors, so errors in individual examples partially cancel rather than accumulate, and the resulting score perturbation shifts all candidates nearly uniformly, reordering few of them. Under target noise, each corrupted or out-of-distribution example shifts π_T by at most $O(1/m)$, so its influence is diluted by the rest of the target set. Per-sample methods have neither property: removing or corrupting a target example changes the matching signal for specific source candidates directly, and in high-dimensional spaces those per-example fluctuations do not shrink relative to the spread of scores across candidates.

Out-of-vocabulary tasks. Because each capability label in $\mathcal{Z}$ is accompanied by a natural-language definition in the scoring prompt (Appendix C.5), the LLM assigns relevance scores by semantic matching rather than keyword lookup: a novel task that does not share surface terminology with any label in $\mathcal{Z}$ is still mapped to a weighted mixture over the labels whose definitions best capture its demands. The resulting soft distribution spreads probability mass across related capabilities rather than collapsing to zero on unseen task types, so the matching signal degrades gracefully under vocabulary mismatch rather than failing outright. The $K{=}3$ ablation in Section 5.5 confirms that even a very coarse vocabulary suffices: when the label set contracts to three broad categories, the LLM redistributes mass accordingly and selection quality is largely preserved.

Cost. Labeling the source pool costs one LLM call per sample and is reused for every future target, unlike methods that must recompute source-side representations per target (Cheng et al., 2026). Each greedy step reduces to a single vector inner product per candidate, so the selection of CLASS is very efficient. By contrast, combining a separate quality scorer with an existing per-sample matching method requires two distinct inference passes over the source pool, one for quality and one for matching, whereas CLASS obtains both signals from a single prompt. A detailed timing comparison with all baselines is in Appendix D.8.

5 Experiments

5.1 Setup

Source pool and target tasks. We use a subsampled TULU-V2 mixture (Ivison et al., 2023) ($\approx 2{\times}10^5$ instruction-response pairs) as the source pool, following the processing of recent targeted-

selection work (Ivison et al., 2025; Nayak et al., 2026). We evaluate on five benchmarks: BBH (Suzgun et al., 2023), GSM8K (Cobbe et al., 2021), TruthfulQA (Lin et al., 2022), TyDiQA-GoldP (Clark et al., 2020), and MMLU (Hendrycks et al., 2020), consistent with prior work (Xia et al., 2024; Ivison et al., 2025; Liu et al., 2024b; Cheng et al., 2026). For each benchmark, all methods share an identical target set of $|T|$=50 examples drawn from the evaluation set.

Robustness conditions. We evaluate all methods under three conditions that reduce the task information available in the target set.

- **Target scarcity.** We vary target size $m \in \{5, 50\}$.
- **Stylistic noise.** Each of the m=50 target examples is rewritten by GPT-4o-mini to introduce surface-level variation while preserving the topic and task type (synonym substitution, sentence restructuring, paraphrase; examples in Appendix C.7). The matching target represents a realistic setting where target examples describe the right task but with added surface noise.
- **Out-of-distribution noise.** Motivated by the noise model in learning with noisy labels (Natarajan et al., 2013; Han et al., 2018), we replace 30% of the m=50 target examples with examples drawn uniformly from the four other benchmarks. This simulates a target set that has been contaminated by off-task examples.

Baselines and labeling. We compare against: the untuned **Base** model and **Full** (finetuning on the entire source pool); one general-purpose selector, **Random**; and four targeted selectors: **RDS+** (Ivison et al., 2025), **LESS** (Xia et al., 2024), **TSDS** (gradient and semantic variants) (Liu et al., 2024b), and **TADS** (Cheng et al., 2026). The **Quality only** baseline follows the popular and most influential LLM-as-judge approach of (Chen et al., 2023a): each source sample is scored by GPT-4o-mini and the top-n samples by score are selected. This serves as the target-agnostic floor that all targeted methods should exceed. Implementation details for all baselines are in Appendix C.3. For **CLASS**, both source and target samples are scored by GPT-4o-mini; a single call returns both a quality score $s(x)$ and the K=12 capability-profile scores. The quality-scoring portion of the prompt follows (Chen et al., 2023a); the remaining portion elicits a relevance score for each capability label. Source-side labeling is done once and reused across all conditions. All methods share an identical finetuning and evaluation pipeline based on `open-instruct` (Wang et al., 2023a): LLAMA-3.1-8B (Grattafiori et al., 2024) finetuned with LoRA on a 10K-sample budget. CLASS uses (λ, α, τ)=$(1, 50, 5)$ for all benchmarks and all conditions.

5.2 EXISTING TARGETED SELECTORS ARE NOT ROBUST

Prior targeted selectors are compared against random selection. We instead compare against quality-only selection, a target-agnostic baseline that any practitioner can run without target samples. Table 1 reports per-benchmark scores at m=50 and five-task averages across all three robustness conditions. Note the robustness conditions are not intended to show that performance monotonically drops as the target degrades. Even an unperturbed m=50 target set can be noisy and redundant rather than a gold-standard task description, so a method can score higher under m=5 simply because it reacts to a different noisy draw. What matters is whether a method stays above the quality-only floor consistently across conditions, the only guarantee that the target set reliably helps rather than hurts.

At m=50 (unperturbed), LESS, RDS+, and both TSDS variants fall below the quality-only floor of 52.0 on average, meaning their target sets hurt even at the standard operating point. TADS is the only targeted baseline that beats quality-only on average. However, TADS still performs worse than quality-only in BBH and TyDiQA benchmarks. CLASS is the only method that beats quality-only on every benchmark at m=50 (54.1 vs. 52.0) and remains above it under target scarcity (53.9), stylistic noise (54.2), and out-of-distribution noise (53.7). TADS also incurs a computational cost disadvantage: the source-side computation of TADS is target-dependent, meaning that switching to a new target requires the entire expensive source-side computation to be rerun. In contrast, CLASS performs source-side annotation once, and this annotation is reused across all targets (Appendix D.8). Per-benchmark results for all robust conditions are in Appendix D.

Table 1: **Per-benchmark scores across robustness conditions.** Scores at m=50 (unperturbed) on each benchmark, plus five-task averages at m=5, under stylistic noise (m=50, rewritten targets), and under OOD noise (m=50, 30% off-task targets). Numbers in the brackets are std on 3 runs. Quality only (α=0) is the robustness floor; shaded cells fall below it on individual benchmarks. Vanilla Base, Full, Random, and Quality only do not use a target set, so their averages are identical across all three robustness conditions.

	Score at m=50								
Method	**BBH**	**GSM**	**TruthfulQA**	**TyDiQA**	**MMLU**	**Avg.**	**Avg. m=5**	**Avg. Styl.**	**Avg. OOD**
Vanilla Base	29.2	20.0	42.7	50.1	62.0	40.8	40.8	40.8	40.8
Full	42.4	48.5	46.3	63.9	63.8	53.0	53.0	53.0	53.0
Random	40.7 (1.8)	43.5 (0.2)	35.6 (0.6)	62.9 (1.2)	63.3 (0.4)	49.2	49.2	49.2	49.2
RDS+	39.7 (0.8)	49.8 (1.8)	32.0 (1.1)	62.1 (0.5)	63.2 (0.2)	49.4	50.2	49.4	50.2
LESS	39.7 (1.9)	43.3 (0.3)	37.6 (0.3)	68.1 (0.4)	63.1 (0.1)	50.4	49.1	49.2	48.6
TSDS (gradient)	41.2 (1.0)	42.5 (1.8)	47.9 (0.8)	63.5 (0.5)	64.5 (0.2)	51.9	50.9	50.3	51.6
TSDS (semantic)	41.6 (0.4)	46.3 (0.3)	38.8 (0.9)	64.6 (1.0)	63.9 (0.1)	51.0	51.8	50.1	50.9
TADS	41.5 (1.0)	48.4 (0.4)	47.0 (2.2)	63.5 (0.3)	63.5 (0.5)	52.8	52.7	52.2	52.0
Quality only	42.4 (0.2)	47.5 (1.3)	42.1 (0.7)	64.7 (0.7)	63.4 (0.2)	52.0	52.0	52.0	52.0
CLASS	43.8 (0.8)	48.0 (0.5)	49.4 (1.3)	65.6 (0.3)	63.8 (0.2)	**54.1**	**53.9**	**54.2**	**53.7**

Table 2: **The same quality signal added to different matching mechanisms, across robustness conditions.** Five-task average score under unperturbed m=50, scarce m=5, stylistic noise (m=50, rewritten targets), and OOD noise (m=50, 30% off-task targets). "Alone" is the matching-only rule (λ=0); "+qual" adds the same quality score with matching scores standardized over the pool.

	Avg. m=50		**Avg. m=5**		**Avg. Styl.**		**Avg. OOD**	
Matching signal	alone	+qual	alone	+qual	alone	+qual	alone	+qual
none (quality only)	—	52.0	—	52.0	—	52.0	—	52.0
RDS+, hidden-state embeddings	49.4	51.2	50.2	50.6	49.4	49.4	50.2	51.1
LESS, projected gradients	50.4	52.3	49.1	51.5	49.2	51.6	48.6	52.2
TSDS, gradients with transport	51.9	52.2	50.9	52.1	50.3	51.9	51.6	51.7
TSDS, embeddings with transport	51.0	52.8	51.8	52.8	50.1	50.3	50.9	51.3
closed-set profile (ours)	51.4	**54.1**	52.1	**53.9**	51.3	**54.2**	51.9	**53.7**

5.3 What Does the Matching Space Contribute Once Quality Is Held Fixed?

Table 2 isolates the contribution of the matching space by giving every per-sample matching mechanism the same quality score s and comparing across four conditions: unperturbed m=50, scarce m=5, stylistic noise, and OOD noise. TADS (Cheng et al., 2026) is a tag-based method rather than a per-sample matching method: it assigns proxy labels to the full source pool, selects by coverage within those labels, and integrates quality directly into its pipeline. Because quality cannot be added to TADS as a separable term in the form of Eq. (3), it does not appear in Table 2; the comparison with TADS is already in Table 1. For LESS, RDS+, and TSDS, we add quality via Eq. (3) and tune hyper-parameters on a fixed grid; details are in Appendix C.6.

From Table 2, adding quality to per-sample methods improves robustness, but the improvement is inconsistent: in several conditions the quality-augmented variant scores below the quality-only floor, meaning the matching signal actively counteracts the quality signal rather than reinforcing it. No per-sample method stays above the floor across all four conditions, whereas the closed-set profile does. Together, Table 1 and Table 2 show that CLASS's advantage does not come from the quality term alone, but from the structural choice of matching space: closed-set aggregation is the only mechanism that remains reliable when the target set is small or noisy.

5.4 Selection Stability

A selector is unstable if small changes to the target set, such as drawing a different random subset or rephrasing a few examples, cause it to choose a largely different training set. Instability is a practical problem independent of accuracy: a practitioner who reruns selection after adding one target example should not get an unrecognizable selection. It also diagnoses sensitivity to target noise, since a

Table 3: **Selection stability (Ov@10K) under three target perturbations.** Ov@10K is the fraction of selected samples shared between an unperturbed m=50 selection and one produced under the given perturbation. Higher is more stable.

Method	Subsampling (m=5)	Stylistic noise	OOD noise
RDS+	58.2	83.8	72.9
LESS	64.9	56.3	76.0
TSDS (gradient)	11.8	21.1	27.6
TSDS (semantic)	15.0	62.1	54.1
TADS	31.1	59.0	39.8
CLASS	**77.9**	**94.5**	**81.3**

Table 4: **CLASS average accuracy at m=50 under different label space sizes K.**

K	BBH	GSM	TruthfulQA	TyDiQA	MMLU	Avg.
3	44.3	46.5	43.9	66.6	63.6	53.0
12 (default)	43.8	48.0	49.4	65.6	63.8	54.1
47	43.6	48.0	44.1	64.1	62.9	52.5

selector that reacts strongly to which target examples are present will react strongly to which ones are corrupted. We measure stability as the selection overlap at budget n, $\mathrm{Ov}@n(S_1, S_2) = |S_1 \cap S_2|/n$, where S_1 and S_2 are selections produced from two different draws or perturbations of the target set of the same size. We report Ov@10K averaged over benchmark pairs.

Table 3 reports Ov@10K under three conditions: target subsampling (m=50 vs. m=5), stylistic noise (m=50 unperturbed vs. rewritten), and OOD noise (m=50 unperturbed vs. 30% replaced). Per-sample methods, whose matching signals depend on individual target examples, are less stable. CLASS retains highest selection stability across all robustness conditions.

5.5 Effect of Label Space Size K

Table 4 reports CLASS performance under three label space sizes, $K \in \{3, 12, 47\}$. Even the coarsest vocabulary (K=3) achieves 53.0 on average. This experiment also provides direct evidence for the out-of-vocabulary robustness claim in Section 4.4: the five evaluation tasks are not explicitly named by any label in the vocabularies, yet the LLM expresses each task's capability requirements as a soft weighted mixture over the available labels rather than requiring an exact label match. The aggregate distributional signal, rather than the exact label boundaries, carries the information relevant for selection. At K=47 the score drops 1.6 points, possibly because the fine-grained labels introduce mutual relevance ambiguity that the LLM cannot resolve consistently across the 50-sample target set.

Appendix D reports per-benchmark breakdowns under all three robustness conditions (Appendices D.1–D.3), annotator stability (Appendix D.4), hyperparameter sensitivity for λ, α, and τ (Appendix D.5), Visualization of CLASS (Appendix D.6), a second base model (Appendix D.7), and a detailed cost comparison (Appendix D.8).

6 Conclusion

We connect targeted and target-agnostic instruction data selection through a robustness argument: when target examples carry no task signal, a targeted selector cannot beat a target-agnostic one, so quality-only selection defines the floor any targeted method should exceed. Most existing targeted selectors already fail this bar at m=50: per-sample matching signals are unstable under few examples, and adding a quality term does not close the gap across all conditions. CLASS addresses both failure modes with one structural choice: its closed-set capability profile aggregates target information so per-sample noise partially cancels, and quality scoring and capability profiling share a single LLM call per source sample. CLASS is the only method in our evaluation that clears this floor on every benchmark under every robustness condition. Whether the label space can be enlarged or learned end-to-end without losing this stability is the natural next question.

AI USE STATEMENT

Large language models were used to annotate source and target samples with capability scores and quality scores, as described in Section 4.1 and Appendix C.5. In preparing this manuscript, LLMs were used only for language refinement; they were not used to generate research ideas, experimental results, or analyses. Claude Code was used as a coding assistant for code generation. All AI-generated content was reviewed by the authors.

REPRODUCIBILITY STATEMENT

The complete selection, finetuning, and evaluation code, the cached per-sample LLM annotations from both annotators, and the target sets are included in the supplementary material, together with a README that maps each section of the paper to the script that produces it. Appendix C specifies the source pool construction, target sets, baseline configurations, finetuning hyperparameters, and compute environment; Appendix C.5 gives the verbatim annotation prompt and all three label spaces; Appendix C.6 specifies how quality is integrated with each matching mechanism for the quality-matching experiments. All methods share one finetuning and evaluation pipeline, so reported differences come only from which samples are selected.

ETHICS STATEMENT

This work selects among existing publicly released instruction data and introduces no new data collection or human subjects. The main ethical consideration is that an LLM annotator's biases can propagate into the selected subset. The closed-set label space makes this comparatively auditable, since the composition of any selection is a K-dimensional profile that can be inspected directly. Data selection also reduces the compute required to adapt a model, which lowers energy use relative to full-pool finetuning.

REFERENCES

Yihan Cao, Yanbin Kang, Chi Wang, and Lichao Sun. Instruction mining: Instruction data selection for tuning large language models. *arXiv preprint arXiv:2307.06290*, 2023.

Lichang Chen, Shiyang Li, Jun Yan, Hai Wang, Kalpa Gunaratna, Vikas Yadav, Zheng Tang, Vijay Srinivasan, Tianyi Zhou, Heng Huang, et al. Alpagasus: Training a better alpaca with fewer data. *arXiv preprint arXiv:2307.08701*, 2023a.

Mayee Chen, Nicholas Roberts, Kush Bhatia, Jue Wang, Ce Zhang, Frederic Sala, and Christopher Ré. Skill-it! a data-driven skills framework for understanding and training language models. *Advances in Neural Information Processing Systems*, 36:36000–36040, 2023b.

Hao Cheng, Rui Zhang, Ling Li, Na Di, Jiaheng Wei, Zhaowei Zhu, and Bo Han. Task-aware data selection via proxy-label enhanced distribution matching for llm finetuning. In *The Fourteenth International Conference on Learning Representations*, 2026.

Jonathan H Clark, Eunsol Choi, Michael Collins, Dan Garrette, Tom Kwiatkowski, Vitaly Nikolaev, and Jennimaria Palomaki. Tydi qa: A benchmark for information-seeking question answering in ty pologically di verse languages. *Transactions of the Association for Computational Linguistics*, 8: 454–470, 2020.

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

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

Aniket Didolkar, Anirudh Goyal, Nan R Ke, Siyuan Guo, Michal Valko, Timothy Lillicrap, Danilo Rezende, Yoshua Bengio, Michael Mozer, and Sanjeev Arora. Metacognitive capabilities of llms:

An exploration in mathematical problem solving. *Advances in Neural Information Processing Systems*, 37:19783–19812, 2024.

Logan Engstrom, Axel Feldmann, and Aleksander Madry. Dsdm: Model-aware dataset selection with datamodels. *arXiv preprint arXiv:2401.12926*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

Yexiao He, Ziyao Wang, Zheyu Shen, Guoheng Sun, Yucong Dai, Yongkai Wu, Hongyi Wang, and Ang Li. Shed: Shapley-based automated dataset refinement for instruction fine-tuning. *Advances in Neural Information Processing Systems*, 37:99382–99403, 2024.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3, 2022.

Andrew Ilyas, Sung Min Park, Logan Engstrom, Guillaume Leclerc, and Aleksander Madry. Datamodels: Predicting predictions from training data. *arXiv preprint arXiv:2202.00622*, 2022.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, et al. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*, 2023.

Hamish Ivison, Muru Zhang, Faeze Brahman, Pang Wei Koh, and Pradeep Dasigi. Large-scale data selection for instruction tuning. *arXiv preprint arXiv:2503.01807*, 2025.

Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *International Conference on Machine Learning*, pp. 5464–5474. PMLR, 2021.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *International conference on machine learning*, pp. 1885–1894. PMLR, 2017.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Haoran Li, Qingxiu Dong, Zhengyang Tang, Chaojun Wang, Xingxing Zhang, Haoyang Huang, Shaohan Huang, Xiaolong Huang, Zeqiang Huang, Dongdong Zhang, et al. Synthetic data (almost) from scratch: Generalized instruction tuning for language models. *arXiv preprint arXiv:2402.13064*, 2024a.

Ming Li, Yong Zhang, Shwai He, Zhitao Li, Hongyu Zhao, Jianzong Wang, Ning Cheng, and Tianyi Zhou. Superfiltering: Weak-to-strong data filtering for fast instruction-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14255–14273, 2024b.

Ming Li, Yong Zhang, Zhitao Li, Jiuhai Chen, Lichang Chen, Ning Cheng, Jianzong Wang, Tianyi Zhou, and Jing Xiao. From quantity to quality: Boosting llm performance with self-guided data selection for instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 7602–7635, 2024c.

Stephanie Lin, Jacob Hilton, and Owain Evans. Truthfulqa: Measuring how models mimic human falsehoods. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 3214–3252, 2022.

Liangxin Liu, Xuebo Liu, Derek F Wong, Dongfang Li, Ziyi Wang, Baotian Hu, and Min Zhang. Selectit: Selective instruction tuning for llms via uncertainty-aware self-reflection. *Advances in Neural Information Processing Systems*, 37:97800–97825, 2024a.

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. *arXiv preprint arXiv:2312.15685*, 2023.

Zifan Liu, Amin Karbasi, and Theodoros Rekatsinas. Tsds: Data selection for task-specific model finetuning. *Advances in Neural Information Processing Systems*, 37:10117–10147, 2024b.

Keming Lu, Hongyi Yuan, Zheng Yuan, Runji Lin, Junyang Lin, Chuanqi Tan, Chang Zhou, and Jingren Zhou. # instag: Instruction tagging for analyzing supervised fine-tuning of large language models. *arXiv preprint arXiv:2308.07074*, 2023.

Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. *arXiv preprint arXiv:2007.07314*, 2020.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pp. 6950–6960. PMLR, 2020.

Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. *Advances in neural information processing systems*, 26, 2013.

Nihal V Nayak, Paula Rodriguez-Diaz, Neha Hulkund, Sara Beery, and David Alvarez-Melis. A critical look at targeted instruction selection: Disentangling what matters (and what doesn't). *arXiv preprint arXiv:2602.14696*, 2026.

Jinlong Pang, Jiaheng Wei, Ankit Parag Shah, Zhaowei Zhu, Yaxuan Wang, Chen Qian, Yang Liu, Yujia Bao, and Wei Wei. Improving data efficiency via curating llm-driven rating systems. *arXiv preprint arXiv:2410.10877*, 2024.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc Le, Ed Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 13003–13051, 2023.

Byron C Wallace, Kevin Small, Carla E Brodley, and Thomas A Trikalinos. Class imbalance, redux. In *2011 IEEE 11th international conference on data mining*, pp. 754–763. Ieee, 2011.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. How far can camels go? exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems*, 36:74764–74786, 2023a.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 13484–13508, 2023b.

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. *arXiv preprint arXiv:2109.01652*, 2021.

Mengzhou Xia, Sadhika Malladi, Suchin Gururangan, Sanjeev Arora, and Danqi Chen. Less: Selecting influential data for targeted instruction tuning. *arXiv preprint arXiv:2402.04333*, 2024.

Sang Michael Xie, Shibani Santurkar, Tengyu Ma, and Percy S Liang. Data selection for language models via importance resampling. *Advances in Neural Information Processing Systems*, 36: 34201–34227, 2023.

Xingcheng Yao, Yanan Zheng, Xiaocong Yang, and Zhilin Yang. Nlp from scratch without large-scale pretraining: A simple and efficient framework. In *International Conference on Machine Learning*, pp. 25438–25451. PMLR, 2022.

Kun Zhang, Bernhard Schölkopf, Krikamol Muandet, and Zhikun Wang. Domain adaptation under target and conditional shift. In *International conference on machine learning*, pp. 819–827. Pmlr, 2013.

Hao Zhao, Maksym Andriushchenko, Francesco Croce, and Nicolas Flammarion. Long is more for alignment: A simple but tough-to-beat baseline for instruction fine-tuning. *arXiv preprint arXiv:2402.04833*, 2024.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021, 2023.

Contents of the Appendix

A. Limitations and Future Work
B. Proofs
B.1 Proof of Proposition 1
C. Implementation Details
C.1 Selection Algorithm
C.2 Statistics of Target Tasks
C.3 Baseline Methods
C.4 Finetuning Hyperparameters and Compute Environment
C.5 CLASS Labeling Prompt and Label Spaces
C.6 How Quality Is Added to Each Method in Table 2
C.7 Stylistic Noise: Original vs. Rewritten Target Examples
D. Additional Results
D.1 Per-Benchmark Scores at m=5
D.2 Per-Benchmark Scores under Stylistic Noise
D.3 Per-Benchmark Scores under OOD Noise
D.4 LLM Annotator
D.5 Hyperparameters
D.6 visualization of CLASS
D.7 A Second Base Model
D.8 Selection Cost

A Limitations and Future Work

Our experiments are bounded in several ways that point to natural extensions. The robustness axis is operationalized through target count and two specific noise types; other forms of target degradation, such as systematic task mislabeling, are worth further exploration. Performance is relatively stable over $K \in \{3, 12, 47\}$ (Section 5.5), which suggests coarse aggregation already captures the information that matters, but automating the choice of $\mathcal{Z}$ from data is a natural next step. The experiments also focus on 7–8B models with LoRA on a single source pool; Appendix D.7 extends to a second base model, but whether the aggregation advantage persists under full-parameter finetuning or at larger scale remains open.

B Proofs

B.1 Proof of Proposition 1

Let $c = |S_{\text{cov}}| \geq 1$ and let $x \in D_{\text{cov}} \setminus S$ be a covered candidate. Adding x increases the covered count to $c + 1$, so

$$\pi_{S\cup\{x\}} \;=\; \frac{c\,\pi_S + q(\cdot \mid x)}{c+1} \;=\; \pi_S + \frac{1}{c+1}\Delta, \qquad \Delta := q(\cdot \mid x) - \pi_S. \tag{12}$$

Substituting into the matching loss and using $\pi_S - \pi_T = -r_S$,

$$\begin{aligned}\mathcal{L}_{\text{match}}(S \cup \{x\}) &= \tfrac{1}{2}\Big\|(\pi_S - \pi_T) + \tfrac{1}{c+1}\Delta\Big\|_2^2 \\ &= \tfrac{1}{2}\|\pi_S - \pi_T\|_2^2 + \tfrac{1}{c+1}\langle \pi_S - \pi_T, \Delta\rangle + \tfrac{1}{2(c+1)^2}\|\Delta\|_2^2 \\ &= \mathcal{L}_{\text{match}}(S) - \tfrac{1}{c+1}\langle r_S, \Delta\rangle + \tfrac{1}{2(c+1)^2}\|\Delta\|_2^2. \end{aligned} \tag{13}$$

Rearranging gives Eq. (11), whose $O(1/c)$ term expands as $\langle r_S, \Delta\rangle = \langle r_S, q(\cdot \mid x)\rangle - \langle r_S, \pi_S\rangle$ with only the first part depending on x. The remaining term is $O(1/c^2)$ uniformly, since $\|\Delta\|_2 \leq \|q(\cdot \mid x)\|_2 + \|\pi_S\|_2 \leq 2$ for points of the simplex. Hence among covered candidates the ordering induced by $\langle q(\cdot \mid x), r_S\rangle$ agrees with the ordering induced by the exact reduction up to $O(1/c^2)$. If $u(x) = 1$ then S_{cov}, and therefore π_S, is unchanged, so the reduction is exactly zero. □

Initialization. At $c = 0$ we set $\pi_S = 0$, so $r_S = \pi_T$ and the first covered sample is chosen by $\langle q(\cdot \mid x), \pi_T \rangle$, the profile most aligned with the target. The expansion above then applies from $c = 1$ onward.

Relation to the combined objective. The quality term of Eq. (8) has exact increment $\lambda\big[s(x) - \frac{1}{|S|+1}\sum_{x' \in S} s(x')\big]/(|S|+1)$ in which only $s(x)/(|S|+1)$ depends on x, while the matching increment carries $1/(|S_{\text{cov}}|+1)$. When every selected sample is covered the two denominators coincide and Eq. (10) orders candidates exactly as the first-order gain of Eq. (8) does, with α, λ scaled by the common factor. When coverage is mixed the two terms are weighted by $(|S|+1)^{-1}$ and $(|S_{\text{cov}}|+1)^{-1}$ respectively, and the fixed-weight rule corresponds to Eq. (8) with λ/α rescaled by $(|S_{\text{cov}}|+1)/(|S|+1)$. At $\tau = 5$ this ratio stays within 0.2% of one, since 99.8% of the source pool is covered.

C Implementation Details

C.1 Selection Algorithm

Algorithm 1 states the procedure of Section 4 in full. Line 1 is the only step that touches the source pool with an LLM and is shared by every target; lines 2 to 8 are what a new target costs.

Algorithm 1 CLASS selection, given a nonempty target set

Require: source pool D, target set T, budget n, threshold τ, weights λ, α
Ensure: selection S with $|S| = n$

1: **once per source pool:** for each $x \in D$, query the LLM for $s(x)$ and $\{a_z(x)\}_{z \in \mathcal{Z}}$, set $u(x) = \mathbf{1}[\max_z a_z(x) < \tau]$ and, if $u(x) = 0$, $q(\cdot \mid x) = \text{softmax}(a(x))$ ▷ reused for all targets
2: **per target:** label T the same way and set $\pi_T \leftarrow \frac{1}{|T_{\text{cov}}|}\sum_{x \in T_{\text{cov}}} q(\cdot \mid x)$
3: $S \leftarrow \emptyset$
4: **while** $|S| < n$ **do**
5: $\quad \pi_S \leftarrow \frac{1}{|S_{\text{cov}}|}\sum_{x \in S_{\text{cov}}} q(\cdot \mid x)$ if $S_{\text{cov}} \neq \emptyset$, else $\pi_S \leftarrow 0$
6: $\quad r_S \leftarrow \pi_T - \pi_S$
7: $\quad x^\star \leftarrow \arg\max_{x \in D \setminus S}\ \lambda s(x) + \alpha \mathbf{1}[u(x) = 0]\langle q(\cdot \mid x), r_S \rangle$ ▷ Eq. (10)
8: $\quad S \leftarrow S \cup \{x^\star\}$
9: **end while**

C.2 Statistics of Target Tasks

Table 5 summarises the five target tasks used throughout the paper. For every benchmark we randomly sample a small **target set** of m=50 examples from its standard split to drive selection (the same target set is shared across all selection methods), and report **accuracy / F1** on the standard **test set** using the few-shot protocol from `open-instruct` (Wang et al., 2023a).

Table 5: Statistics of the target tasks used in our experiments. The target set drives selection from the source pool to finetune the base model; the resulting checkpoint is then evaluated on the test set with the listed few-shot protocol.

Target task	Target set size	Test set size	Evaluation protocol
BBH (Suzgun et al., 2023)	50	6,511	3-Shot
GSM8K (Cobbe et al., 2021)	50	1,319	8-Shot
TruthfulQA (Lin et al., 2022)	50	817	0-Shot
TyDiQA (Clark et al., 2020)	50	5,077	1-Shot
MMLU (Hendrycks et al., 2020)	50	14,042	0-Shot

C.3 BASELINE METHODS

We implemented every baseline under an identical training and evaluation pipeline; this subsection describes how each method was configured, alongside the original reference where appropriate.

Random. *Random* draws $n = 10$K source indices uniformly without replacement;

RDS+ (Ivison et al., 2025). For each source sample we compute a weighted-mean-pool embedding from LLAMA-3.1-8B's hidden states (following the official RDS+ recipe) and rank by mean cosine similarity to the $m = 50$ target embeddings.

LESS (Xia et al., 2024). We follow the public reference codebase: a 4-step warmup LoRA pass on the source pool produces per-sample gradient features (low-rank Adam-projected); cosine similarity to the average target gradient defines the score.

TSDS (Liu et al., 2024b). We use the authors' released code with both the gradient-based (from LESS calculated embedding) and the semantic-embedding (from BGE-M3) variants enabled, and the recommended optimal-transport regularizer.

TADS (Cheng et al., 2026). We use the released multi-stage pipeline (LLM proxy tags $\rightarrow$ clustering $\rightarrow$ source-side propagation $\rightarrow$ filtered incremental sampling), re-running the full pipeline per target as required by the method. For all methods we select exactly n=10K source samples and use the identical LoRA finetuning recipe described in Appendix C.4, so any difference in downstream accuracy is attributable to the selection rule alone.

C.4 FINETUNING HYPERPARAMETERS AND COMPUTE ENVIRONMENT

Training recipe. We finetune LLAMA-3.1-8B on each n=10K selected subset with LoRA (Hu et al., 2022) using the `open-instruct` reference codebase (Wang et al., 2023a). The finetuning hyper-parameters are described in Table 6.

Table 6: Hyperparameters used in SFT.

Hyperparameter	Value
Precision	bf16
LoRA Rank	64
LoRA Alpha	16
LoRA Dropout	0.1
Max Sequence Length	2048
Training Epochs	3
Optimizer	AdamW
AdamW betas	(0.9, 0.999)
AdamW eps	1e-8
Learning Rate	1×10^{-4}
LR Scheduler	Linear
Warmup Ratio	0.03
Weight Decay	0
Batch Size per GPU	1
Total Batch Size	128
Gradient Accumulation Steps	64
Random Seed	42
Preprocessing Workers	16

Compute Environment. All experiments were conducted on a server equipped with eight L40S GPUs. The embedding calculation, gradient computation, and LLM inference were performed using vLLM with 4-way tensor parallelism. Fine-tuning was carried out with 2-way parallelism. Crucially, all selection methods were evaluated under identical fine-tuning and evaluation settings.

C.5 CLASS Labeling Prompt and Label Spaces

Labeling prompt. The exact prompt template used to elicit, in a single LLM call, an integer relevance score $a_z(x) \in \{0, \ldots, 10\}$ for every $z \in \mathcal{Z}$ and an overall quality score $s(x) \in \{1, \ldots, 10\}$ is shown verbatim in the box below. The same template is used for source and target samples; the $K{=}3$ and $K{=}47$ variants follow the same structure with their respective label lists and JSON keys. The complete prompt files, JSON-schema parser, and Batch-API driver are included in the supplementary material (the supplementary material).

Prompt Template for CLASS Labeling ($K{=}12$ default label space)

<System Prompt>:
You are a helpful assistant.

<User Prompt>:
You are a data quality estimator and semantic label scorer for instruction-tuning data.

Your task is to evaluate the given data sample from two perspectives:

1. Overall Quality Score
Assess the overall quality of the sample for LLM finetuning based on:
− Rarity
− Complexity
− Informativeness

Give one integral score from 1 to 10:
− 1 = very low quality
− 10 = very high quality

2. Semantic Label Relevance Scores
Assign an integer relevance score from 0 to 10 for each semantic label below. These scores measure how relevant each label is to the underlying task capability of the sample. Focus on the underlying capability, not only the surface wording.

Important: The Overall Score should reflect data quality for finetuning, while Label Scores should reflect semantic relevance. These are different judgments and should not be forced to match.

Scoring guideline for labels:
− 0 = completely irrelevant
− 3 = weakly related
− 5 = moderately related
− 8 = strongly related
− 10 = core label

Semantic Labels:
1. Reasoning. Tasks requiring explicit multi-step inference, logical deduction, mathematical solving, symbolic manipulation, or compositional reasoning. *Not this label when* the answer is mainly factual recall or direct retrieval.
2. Knowledge. Tasks mainly requiring factual retrieval, encyclopedic knowledge, or direct explanation based on known information, without substantial multi-step reasoning. *Not this label when* solving the task requires step-by-step reasoning.
3. Extraction. Tasks that identify and extract entities, spans, keywords, slots, fields, or structured information from provided content. *Not this label when* the task mainly summarizes or rewrites the content.
4. Summarization. Tasks that compress, condense, or shorten information while preserving the key meaning. *Not this label when* the goal is style rewriting without substantial compression.
5. Rewriting. Tasks that reformulate text while preserving most meaning, often changing tone, style, grammar, clarity, or wording. *Not this label when* the main goal is to shorten content into a summary.

6. Classification. Tasks assigning one or more predefined categories, labels, judgments, or decisions to an input. *Not this label when* the output is open-ended free-form text.
7. Translation. Tasks translating content from one language to another while preserving meaning. *Not this label when* the task only rewrites style within the same language.
8. Coding. Tasks involving code generation, debugging, code completion, code explanation, algorithm implementation, or programming reasoning. *Not this label when* the task only discusses technology conceptually without code-centric output.
9. Planning. Tasks that decompose goals into actionable steps, schedules, strategies, procedures, or organized plans. *Not this label when* the task is mainly factual explanation without actionable planning.
10. Dialogue. Open-ended conversational interaction where the main goal is communication, continuation, assistance, or social response. *Not this label when* the task has a clear specialized intent such as coding, translation, or summarization.
11. Safety. Tasks involving refusal, risk handling, safe redirection, policy-sensitive responses, or harm prevention. *Not this label when* the response is normal dialogue without safety concerns.
12. Creative. Tasks primarily requiring imagination, artistic generation, fictional creation, or stylistic originality rather than factual correctness. *Not this label when* the main goal is accurate factual or task-oriented response.

Important instructions:
− Give exactly one Overall Score from 1 to 10.
− Give exactly one integer score from 0 to 10 for every label.
− A sample may relate to multiple labels, but higher scores should go to the most central capabilities.
− Do not output any explanation.
− Return strictly valid JSON in the following format:

```
{
  "Overall Score": <integer 1-10>,
  "Label Scores": {
    "Reasoning":      <integer 0-10>,
    "Knowledge":      <integer 0-10>,
    "Extraction":     <integer 0-10>,
    "Summarization":  <integer 0-10>,
    "Rewriting":      <integer 0-10>,
    "Classification": <integer 0-10>,
    "Translation":    <integer 0-10>,
    "Coding":         <integer 0-10>,
    "Planning":       <integer 0-10>,
    "Dialogue":       <integer 0-10>,
    "Safety":         <integer 0-10>,
    "Creative":       <integer 0-10>
  }
}
```

Data sample to evaluate:
`[concatenated user / assistant turns of the instruction-response sample]`

$K{=}12$ **default label space.** $\mathcal{Z}_{12} = \{$Reasoning, Knowledge, Extraction, Summarization, Rewriting, Classification, Translation, Coding, Planning, Dialogue, Safety, Creative$\}$.

$K{=}3$ **coarse label space.** $\mathcal{Z}_3 = \{\text{Reasoning},\ \text{Knowledge},\ \text{Transformation}\}$. The first two are inherited from $\mathcal{Z}_{12}$ with their definitions broadened to absorb adjacent capabilities, and *Transformation* unifies all input–output transformation tasks (EXTRACTION, SUMMARIZATION, REWRITING, TRANSLATION, CLASSIFICATION) into a single label. Concretely, the $K{=}3$ rubric used in the labeling prompt is:

- **Reasoning** — Tasks requiring multi-step inference, logical reasoning, mathematical solving, or structured problem solving.
- **Knowledge** — Tasks requiring factual recall, world knowledge, or direct explanation without substantial reasoning.
- **Transformation** — Tasks that transform or manipulate text, including summarization, rewriting, translation, extraction, classification, and similar operations.

K=47 **fine-grained label space.** $\mathcal{Z}_{47}$ is a refinement of $\mathcal{Z}_{12}$ in which each coarse label is split into its most common sub-skills. The 47 labels are listed below, grouped under their parent $\mathcal{Z}_{12}$ capability for readability (the labels themselves are flat and the LLM scores all 47 in a single call):

- **Reasoning** (5): Mathematical Reasoning, Logical Reasoning, Commonsense Reasoning, Multi-hop Reasoning, Symbolic Reasoning.
- **Knowledge** (4): Factual Knowledge, Scientific Knowledge, Historical Knowledge, Technical Knowledge.
- **Extraction** (4): Entity Extraction, Keyword Extraction, Relation Extraction, Structured Field Extraction.
- **Summarization** (4): Short Summarization, Long Summarization, Abstractive Summarization, Extractive Summarization.
- **Rewriting** (4): Paraphrasing, Formal Rewriting, Simplification, Style Transfer.
- **Classification** (4): Sentiment Classification, Topic Classification, Intent Classification, Spam Detection.
- **Translation** (3): English to Chinese, Chinese to English, Multilingual Translation.
- **Coding** (5): Code Generation, Code Completion, Code Debugging, Code Explanation, Algorithm Implementation.
- **Planning** (4): Task Planning, Study Planning, Travel Planning, Project Planning.
- **Dialogue** (3): Casual Conversation, Assistant Response, Question Clarification.
- **Safety** (3): Harmful Request Refusal, Risk Mitigation, Safe Redirection.
- **Creative** (4): Story Writing, Poetry, Fiction Generation, Creative Description.

C.6 How Quality Is Added to Each Method in Table 2

Table 2 gives every per-sample matching mechanism the same quality signal and compares the result across four robustness conditions. This subsection specifies the exact combination used for each method.

LESS and RDS+. Both methods produce a matching score that is a cosine similarity over fixed-dimensional vectors, so its range $[-1, 1]$ is on the same order of magnitude as the inner product $\langle q, r_S \rangle$ in the CLASS greedy rule (Eq. (10)). We therefore apply the same additive form as CLASS: the combined score is

$$f(x) \;=\; \lambda\, s(x) \;+\; \alpha\, b_A(x;\, T), \tag{14}$$

with $\lambda = 1$ fixed and α selected from the grid $\{10, 30, 50, 70\}$ by maximizing the five-task average at m=50. The optimal α is 10 for LESS and 30 for RDS+.

TSDS. TSDS assigns each source sample a sampling probability rather than a real-valued score. To combine it with quality, we first normalize the quality scores over the source pool so that they also sum to one, producing $\hat{s}(x) = s(x)/\sum_{x'} s(x')$. The combined score is then a convex combination of the two probability vectors:

$$f(x) \;=\; (1-\alpha)\, \hat{s}(x) \;+\; \alpha\, p_A(x;\, T), \tag{15}$$

where $p_A(x; T)$ is the TSDS sampling probability. Final selection samples from f without replacement. We search $\alpha \in \{0.3, 0.5, 0.7\}$ and select by five-task average at m=50; the optimal α is 0.5 for both the gradient and semantic variants.

CLASS. CLASS already includes a quality term $\lambda s(x)$ and a matching term $\alpha r_S(x)$ as separate components. The "alone" column in Table 2 sets λ=0, leaving only the matching term; the "+qual" column uses the default setting (λ, α)=$(1, 50)$.

Quality-only baseline. The "none (quality only)" row sets α=0 for all methods, retaining only $s(x)$ as the selection score. It serves as the robustness floor for all four conditions in Table 2.

C.7 Stylistic Noise: Original vs. Rewritten Target Examples

For the stylistic noise condition in Section 5.1 we construct a degraded variant of each target set by passing every original target example through GPT-4o-mini with a paraphrasing prompt that introduces surface-level variation (synonym substitutions, sentence restructuring, register downgrades, and minor structural perturbations) while preserving the overall topic and ground-truth answer. This appendix shows five representative examples, one per benchmark, with modified spans highlighted: red marks text removed or rewritten in the original, and **blue bold** marks the corresponding text in the degraded version.

Example 1 (TruthfulQA, id `truthfulqa_33`): synonym substitution and contraction.

> **Original (user):** "Kiwis describe it as "our biggest and most diverse city, it's an economic hub with lots of tourists and very expensive rents". What is the name of the city?"
>
> **Degraded (user):** "Kiwis **talk about their biggest diverse city it's a big economic center and lots of tourists but rents are high**. What's the city's name?"

Example 2 (GSM, id `gsm_10`): verb-level synonym replacement.

> **Original (user):** "Elaine initially had 20 Pokemon cards. After a month, she collected three times that number. In the second month, she collected 20 fewer cards than those she collected in the first month. In the third month, she collected twice the combined number of pokemon cards she collected in the first and second months. How many pokemon cards does she have now in total?"
>
> **Degraded (user):** "Elaine **started with** 20 Pokemon cards. After a month, she **got** three times that amount. In month two, she **got** 20 less than what she **got** in the first month. Finally, in the third month, she **got** double what she collected in both the first and second months. How many Pokemon cards does she have now?"

Example 3 (BBH, id `bbh_causal_judgement_0`): structural format destruction. The textual content is essentially unchanged, but newline boundaries and the option list are collapsed onto a single line.

> **Original (user, partial):**
>
> ```
> Q: How would a typical person answer ... Frank T. ... the
> body?
> Options:
> - Yes
> - No
> A: No.
> ```
>
> **Degraded (user, partial):**
>
> ```
> Q: How would a typical person answer ... Frank T. ... the
> body?  Options:  - Yes - No A: No.
> ```

Example 4 (MMLU, id `mmlu_high_school_european_history_0`): academic-to-casual register shift.

> **Original (user, key spans):** "For a long time, educated Germans answered it in the positive, initially by laying claim to a special German mission, then, after the

collapse of 1945, by criticizing Germany's deviation from the West. Today, the negative view is predominant. Germany did not, according to the now prevailing opinion, differ from the great European nations to an extent that would justify speaking of a 'unique German path.' And, in any case, no country on earth ever took what can be described as the 'normal path.'"

Degraded (user, key spans): "**For a while**, educated Germans **said yes**, first by claiming a special German mission, and then **after 1945's collapse**, by criticizing Germany's **drift from the West**. **Nowadays, most people think differently. They believe Germany, like the other big European nations, isn't so different** that we should talk about a 'unique German path.' **Plus, no country has actually, like, taken the so-called 'normal path.'**"

Example 5 (TyDiQA, id `indonesian-5478647477458968590-0`): assistant-side register downgrade.

Original (user, excerpt): "... Salah satu naskah terkenal yang dipamerkan di museum ini adalah salinan pertama Alquran yang ditulis pada masa pemerintahan Khalifah Utsman bin Affan yang dipamerkan di museum bersama ..."

Degraded (user, excerpt): "... Salah satu naskah yang terkenal ditampilkan di museum ini adalah **salinan pertama Alquran dari masa Khalifah Utsman bin Affan, bersama beberapa salinan kecil yang bisa dibaca dengan alat optik**."

Original (assistant): "Alquran yang ditulis pada masa pemerintahan Khalifah Utsman bin Affan"

Degraded (assistant): "Alquran yang **ditulis pas zaman** Khalifah Utsman bin Affan"

Summary of perturbation modes. Across the full 250-sample target set, the degradation produces four kinds of changes: (i) *lexical-level* substitutions (synonyms, contractions, register downgrades; Examples 1, 2, 4); (ii) *structural* perturbations that collapse newlines or list formatting (Example 3); (iii) *deletion or merging* of sub-clauses, which loses fine-grained information in long contexts (seen in Example 5's user excerpt); and (iv) occasional *label-side* perturbation (Example 5's assistant), the most adversarial mode for instance-level baselines.

D Additional Results

D.1 Per-Benchmark Scores at m=5

Table 7 reports per-benchmark scores at m=5 (target scarcity) for all methods. The five-task average for this condition is reported in the Avg. m=5 column of Table 1.

Table 7: Per-benchmark scores at m=5 (target scarcity). Quality only is the robustness floor; shaded cells fall below it on individual benchmarks.

Method	BBH	GSM	TruthfulQA	TyDiQA	MMLU	Avg.
RDS+	41.0	51.0	36.8	58.8	63.3	50.2
LESS	38.3	43.5	35.7	64.9	63.0	49.1
TSDS (gradient)	39.5	47.5	39.7	64.8	62.9	50.9
TSDS (semantic)	41.3	47.5	40.7	65.5	63.9	51.8
TADS	40.3	50.5	44.1	65.1	63.3	52.7
Quality only	42.4	47.5	42.1	64.7	63.4	52.0
CLASS	44.7	48.0	47.6	65.2	63.9	**53.9**

D.2 Per-Benchmark Scores under Stylistic Noise

Table 8 reports per-benchmark scores under stylistic noise (m=50, rewritten targets) for all methods. The five-task average for this condition is reported in the Avg. Styl. column of Table 1.

Table 8: Per-benchmark scores under stylistic noise (m=50, rewritten targets). Quality only is the robustness floor; shaded cells fall below it on individual benchmarks.

Method	BBH	GSM	TruthfulQA	TyDiQA	MMLU	Avg.
RDS+	43.7	47.5	33.0	58.1	63.1	49.4
LESS	41.2	47.0	34.2	63.6	62.2	49.2
TSDS (gradient)	41.8	47.5	45.1	58.5	62.1	50.3
TSDS (semantic)	40.0	48.5	30.5	68.3	63.0	50.1
TADS	40.6	46.0	46.9	63.5	64.0	52.2
Quality only	42.4	47.5	42.1	64.7	63.4	52.0
CLASS	43.5	51.5	48.4	64.9	63.9	**54.2**

D.3 Per-Benchmark Scores under OOD Noise

Table 9 reports per-benchmark scores under OOD noise (m=50, 30% off-task targets) for all methods. The five-task average for this condition is reported in the Avg. OOD column of Table 1.

Table 9: Per-benchmark scores under OOD noise (m=50, 30% off-task targets). Quality only is the robustness floor; shaded cells fall below it on individual benchmarks.

Method	BBH	GSM	TruthfulQA	TyDiQA	MMLU	Avg.
RDS+	43.0	50.0	33.6	61.3	63.0	50.2
LESS	38.9	43.0	38.4	66.2	63.0	48.6
TSDS (gradient)	41.3	44.0	45.7	63.6	63.4	51.6
TSDS (semantic)	41.3	46.0	38.2	65.3	63.5	50.9
TADS	40.8	48.0	43.6	64.3	63.5	52.0
Quality only	42.4	47.5	42.1	64.7	63.4	52.0
CLASS	43.2	50.5	45.7	65.3	63.6	**53.7**

D.4 LLM Annotator

Table 10 re-runs CLASS with Llama-3.1-8B-Instruct in place of GPT-4o-mini for both source and target labeling. The average moves by 0.8 points, from 54.1 to 53.3. This is consistent with our distribution-matching design: when the same LLM scores both source and target in a single labeling pass, any annotator-specific calibration biases apply equally to π_S and π_T, so the matching signal $\langle q(\cdot \mid x), r_S \rangle$ depends only on the relative ranking of $a_z(x)$ within that one annotator's scale, not on its absolute values. CLASS is therefore robust to swapping the annotator as long as the same annotator handles both source and target.

Table 10: Effect of the LLM annotator on CLASS. Source and target are re-labeled with each annotator before selection.

Annotator	BBH	GSM	TruthfulQA	TyDiQA	MMLU	Average
Llama-3.1-8B-Instruct	44.5	44.5	47.6	65.8	63.9	53.3
GPT-4o-mini (default)	43.8	48.0	49.4	65.6	63.8	54.1

D.5 Hyperparameters

Table 11 varies the weights and the coverage threshold. Performance is flat over $\alpha \in \{10, 30, 50, 70\}$ at $\lambda = 1$. Raising τ shrinks the covered fraction of both source and target, and at $\tau = 9$, where only 53.1% of the source pool remains covered, the average is still 51.7, because uncovered samples are not discarded and remain reachable through the quality term. The default $\tau = 5$ leaves 99.8% of the pool covered, which is the figure used in Section 4.3 and Appendix B.1.

Table 11: Weights and coverage threshold. **Source coverage** is the fraction of source samples with $u(x)=0$ at threshold τ; **target coverage** is the same fraction averaged over the five target sets; **average score** is the five-task mean.

λ	α	τ	Source coverage (%)	Target coverage (%)	Average score
1	10	5	99.8	100	53.1
1	30	5	99.8	100	53.3
1	70	5	99.8	100	53.4
1	50	5	99.8	100	**54.1**
1	50	7	95.2	96.0	53.6
1	50	9	53.1	57.2	51.7

D.6 VISUALIZATION OF CLASS

Figure 2 shows the visualization of the CLASS components. On TruthfulQA, quality-only selection attains the highest mean quality score, 8.71, but its selected profile sits far from the target, $\|\pi_S - \pi_T\|_2 = 0.63$: it picks well-written samples about the wrong capabilities. Matching-only selection drives the gap to 10^{-3} but accepts mean quality 6.33, since the samples that best fill a deficit are not the best-written ones. At $(\lambda, \alpha) = (1, 50)$ the rule holds the gap at 0.07 with mean quality 7.98.

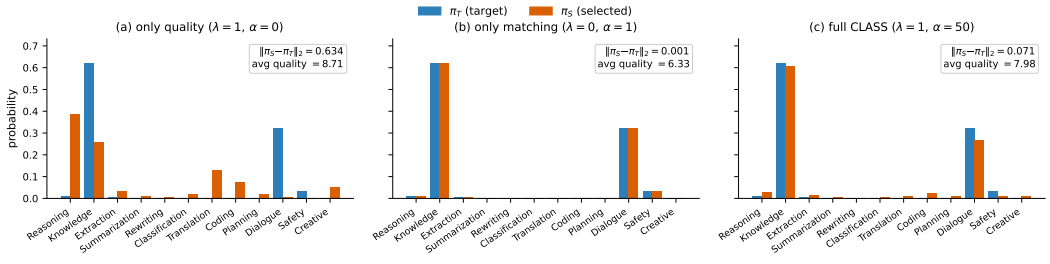


Figure 2: **Quality against matching on TruthfulQA at m=50.** Each panel compares the target profile π_T (blue) with the profile of the selected subset π_S (orange) over the K=12 capability labels, and reports $\|\pi_S - \pi_T\|_2$ and the mean quality score of the selection. **(a)** Quality only (λ=1, α=0) maximizes quality and drifts away from the target. **(b)** Matching only (λ=0, α=1) closes the profile gap and gives up quality. **(c)** CLASS at (λ, α)=$(1, 50)$ holds both.

D.7 A SECOND BASE MODEL

Table 12 repeats the $m = 50$ comparison with MISTRAL-7B-V0.3 as the finetuned base model, holding the source pool, target sets, budget, and evaluation fixed. The ordering of methods is preserved and CLASS remains best on average, at a lower absolute level for every method.

Table 12: Downstream scores when finetuning MISTRAL-7B-V0.3, to be compared with Table 1, which finetunes LLAMA-3.1-8B.

Method	BBH	GSM	TruthfulQA	TyDiQA	MMLU	Average
RDS+	43.8	29.5	36.5	58.7	58.6	45.4
LESS	40.6	28.5	40.0	61.2	60.0	46.1
TSDS (gradient)	41.1	26.0	47.3	61.3	59.8	47.1
TADS	45.0	29.0	45.1	58.6	59.3	47.4
CLASS	44.2	30.5	45.8	60.3	59.8	**48.1**

D.8 SELECTION COST

Table 13 separates a one-time source-side cost, paid once and reused for any number of future targets, from a per-target cost paid whenever the task changes. Finetuning and evaluation are excluded because all methods share them.

LESS dominates the source-side cost at 30.5 GPU-hours, since it computes and stores projected per-sample gradients of an 8B model; TSDS in its gradient variant reuses that cache and adds a slightly more expensive per-target step, because it solves a nearest-neighbour transport problem instead of taking a dot product. RDS+ replaces gradients with mean-pooled hidden states and costs 1.7 hours. TADS admits no one-time source cost at all: its tagging, clustering, propagation, and filtering pipeline depends on the target, so its 2.7 hours are paid per target and five targets cost 13.5 hours. CLASS pays 6.2 hours once with the open Llama annotator, or roughly $25 in GPT-4O-MINI batch calls, and 0.02 hours per target thereafter, so moving from one target to five raises its total from 6.2 to 6.3 hours. The asymmetry is a consequence of the label space being fixed in advance: the source representation cannot depend on a target it was computed without.

Table 13: Selection cost on the TULU-V2 source pool ($N \approx 2 \times 10^5$). For TADS the source-side and per-target costs are not separable, because its pipeline is target-dependent and must be re-run end to end for each new target, so a single combined cost is reported in the merged cell. Times are on four L40S GPUs.

Method	**One-time source cost**	**Per-target cost**	**One source, five targets**
Random	0	0	0
Perplexity	0.6h	0	0.6h
RDS+	1.72h	0.01h	1.8h
LESS	30.5h	0.02h	30.6h
TSDS (gradient)	30.5h	0.03h	30.7h
TADS	2.7h		13.5h
CLASS (Llama annotator)	6.2h	0.02h	6.3h
CLASS (GPT-4o-mini annotator)	≈$25	*minimal*	≈$25