PairDistill: Amortizing Test-Time Pairwise LLM Oracles for Code Selection via Pairwise Judgment Distillation

Anonymous authors
Paper under double-blind review

Abstract

Pairwise LLM oracles can select the best candidate from a set of sampled programs: ExPairT-LLM poses membership and equivalence queries to a frontier LLM at test time and attains strong pass@1, but pays the cost of dozens of frontier-model queries per task at inference time. We conduct a critical analysis of distilling such pairwise oracles into compact trained verifiers. As the analysis vehicle, we build PairDistill, which amortizes this oracle into a compact trained verifier: a frontier teacher (Qwen3-8B) annotates pairwise queries once during training; because code differentiating inputs are executable, we score teacher labels against ground-truth execution outcomes before distillation (execution-scored distillation); a 1.5B student is then trained on the teacher's judgments with swap augmentation and answer-token loss masking. At inference the verifier replaces teacher generation with a single forward pass per query (a 5.3× smaller model, no decoding), using 8 verifier queries per task. On HumanEval+MBPP and TACO with Qwen2.5-3B candidates, the 1.5B verifier improves pass@1 over random selection by +3.2 (HumanEval+MBPP, borderline), +2.4 (TACO-fn) and +1.4 (TACO-stdin) points, recovering 13–15% of the oracle–random gap (23% with a 3B verifier) at 14% of the teacher's per-task query count (8 vs. ∼56 queries; under 8% in token terms, since each student query is a single 1.5B forward pass over ∼4.1K tokens while each teacher query requires 8B generation of a differentiating input and verdict), and exceeds the teacher's own test-time selection quality (64.8% teacher vs. 67.7% student on the full HumanEval+MBPP evaluation; paired Δ +2.9 student−teacher, CI $[-0.5, +6.3]$). Gains over random are statistically significant on both TACO benchmarks (paired bootstrap, 95% CI). Ablations show that distillation volume matters far less than dev metrics suggest (25% of data recovers most selection quality), that verifier scale exhibits a threshold effect (0.5B verifiers match random selection despite higher dev accuracy than 1.5B), that task-level leakage between teacher annotation and evaluation inflates apparent selection gains by several points, and — via a 2×2 de-confound — that the execution-filtering step itself is not responsible for end-to-end gains: under a unified training schedule, the class prior (not label filtering) is the largest axis of variation, and the headline configuration (filtered + balanced) is the only cell whose selection gain over random is individually significant, so the distillation recipe with a balanced query prior carries the result.

1 Introduction

LLM code generation improves substantially when the model samples multiple candidates and selects the best one. Selection without ground-truth tests is the bottleneck: execution-based methods (CodeT (Chen et al., 2023), B^4 (Chen et al., 2024)) require trustworthy test suites, and LLM-as-a-judge approaches query a frontier model per candidate at test time. ExPairT-LLM (Yuviler & Drachsler-Cohen, 2025) frames selection as exact learning with pairwise membership ("is candidate x correct?") and equivalence ("do x, y have the same I/O behavior?") queries to an LLM oracle, achieving state-of-the-art selection. How-

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.

ever, every test task requires ∼56 oracle queries to a frontier LLM, which is expensive at deployment.

We observe that the oracle's judgments are stable across tasks: given the same query pattern (two programs + a differentiating input), the correct verdict is a learnable function. This suggests amortizing the oracle: query the teacher once on training tasks, distill its pairwise judgments into a compact verifier, and answer all future queries with the student. Two obstacles arise. First, teacher judgments are noisy — our teacher agrees with execution ground truth on only 83.6% of queries. Second, pairwise judgments on code have a structure that generic judgment distillation (JudgeLM (Zhu et al., 2025), PairJudge RM (Liu et al., 2025)) does not exploit: differentiating inputs are executable, so the ground truth of each query can be checked.

PairDistillexploits this structure with three components: (1) Pairwise query distillation: we replicate ExPairT-LLM's query protocol (membership + equivalence with differentiating inputs) with a Qwen3-8B teacher on training tasks, yielding 23.7K labeled queries. (2) Execution-scored distillation: every query's ground truth is computed by execution; we use this both to measure teacher quality (16.4% disagreement) and to study what the student should learn from — the teacher's agreement region (filtered labels) or its full output distribution (raw labels). (3) Verifier-based selection: the 1.5B student supports tournament selection over equivalence clusters and a simpler membership-argmax baseline, at 8–21 queries per task instead of ∼56 frontier queries.

Contributions.

1. We present a critical de-confound of what makes distilled pairwise verifiers work, using a 2×2 label-mode × class-prior design under a unified training schedule: filtering improves ranking quality (AUROC 0.861 vs. 0.768 within the balanced column) but not end-to-end selection (+1.7, n.s.) — the class prior, not label curation, is the larger axis, and the distillation recipe with a balanced query prior carries the result.
2. We characterize when distilled verifiers help: a scale-threshold finding (0.5B students reach near-parity pairwise accuracy yet fail to beat random selection) shows selection quality is more sensitive than pairwise metrics, and an oracle-headroom analysis shows gains scale with the random–oracle gap.
3. As the analysis vehicle, we build PairDistill, a framework that distills pairwise equivalence/membership judgments into a compact trained verifier for code selection, using execution-scored distillation (enabled by the executable-verifiable ground truth of code queries); the resulting 1.5B verifier matches or exceeds its teacher's test-time selection quality while replacing 8B generation with single forward passes.
4. We compare four selection strategies under a unified evaluation and find membership-argmax matches or exceeds tournament selection at a third of the cost (67.7 vs. 66.3 pass@1, paired Δ +1.4, CI $[-0.8, +3.6]$, n.s.; 8.0 vs. 20.7 queries on HumanEval+MBPP).

2 Related Work

Pairwise LLM oracles for code selection. ExPairT-LLM (Yuviler & Drachsler-Cohen, 2025) poses membership and equivalence queries to an LLM at test time within an exact-learning framework. CodeT (Chen et al., 2023) and B^4 (Chen et al., 2024) cluster candidates by execution; SRank (To et al., 2024) reranks by functional overlap; SEP (Cho et al., 2026) partitions candidates via symbolic execution. All require execution at selection time; PairDistillreplaces it with a trained verifier.

Distilling LLM judgments. JudgeLM (Zhu et al., 2025) fine-tunes judges on GPT-4 comparisons of open-ended text; PairJudge RM (Liu et al., 2025) distills 432K pairwise correctness judgments for math Best-of-N via knockout tournaments; Prometheus 2 (Kim et al.,

Algorithm 1 PairDistill: amortizing a test-time pairwise oracle into a compact verifier.

Require: Training tasks; frontier teacher T; candidate generator G; student S
Ensure: A trained verifier S and a selection protocol

1: Annotate: for each task, sample n candidates from G, execute on reference tests, and form membership/equivalence queries (each equivalence query carries a differentiating input d found by execution); collect teacher verdicts
2: Score: compute ground truth $\mathrm{gt}(q)$ for every query by executing the candidates on d
3: Curate: keep queries where T agrees with $\mathrm{gt}(q)$ (filtered) or keep all (raw); rebalance the positive/negative prior
4: Distill: fine-tune S with LoRA on the curated queries, applying swap augmentation and answer-token loss masking
5: Select: at inference, score candidates by p_{yes} and pick c^* via membership-argmax, tournament, or adaptive early-exit
6: return c^*

2024) and PandaLM (Wang et al., 2023) train open evaluators on preference data. PairDistill (Huang & Chen, 2024) — homonymous with our method but different in task and mechanism — distills pairwise relevance judgments for dense retrieval. None of these targets program equivalence, and none can filter supervision by execution. ELSPR (Yu et al., 2025) cleans non-transitive preference data via tournament graphs — a complementary, execution-free alternative to our execution filter.

Trained code verifiers and rankers. CodeRanker (Inala et al., 2022) and RankEF (Sun et al., 2024) train pointwise fault predictors; Themis (Paul et al., 2026) trains code reward models down to 600M parameters from preference data; CodeRM-8B (Ma et al., 2025) trains unit-test generators. CETBench (Oza et al., 2025) and EquiBench (Wei et al., 2025) benchmark LLM program-equivalence checking and show frontier models remain brittle, motivating distillation from an execution-checked teacher. ProbeGen (Allamanis & Yin, 2025) disproves equivalence with generated counterexamples at test time.

3 Method

Algorithm 1 summarizes the end-to-end pipeline; the rest of this section details each stage.

3.1 Problem and query types

Given a prompt p and n sampled candidates $C = \{c_1, \ldots, c_n\}$, select c^* maximizing the probability of passing all hidden tests. Following ExPairT-LLM, a verifier may ask two query types about candidates x, y: membership "does x pass p's tests?" and equivalence "do x and y have identical I/O behavior on p?" Equivalence queries come with a differentiating input d; the query includes d so the answer is determined by $(x(d), y(d))$.

3.2 Teacher annotation

For each training task we sample n=8 candidates from the student generator (Qwen2.5-3B, base), execute them on the reference tests, and construct queries: membership queries for each candidate; equivalence queries for candidate pairs, each paired with a differentiating input (an input where the two programs' outputs differ, found by execution; pairs with no differentiating input are labeled equivalent on the input space we can probe). The teacher (Qwen3-8B) answers each query in a fixed yes/no format. This yields 23,674 queries over 3,249 tasks. All teacher annotation uses TACO tasks only; HumanEval and MBPP are reserved exclusively for selection evaluation, making those results zero-shot cross-dataset transfer (§4).

3.3 Execution-scored distillation

Each query's ground truth is computable by executing the candidates on the relevant inputs. We compute gt(q) for every query and use it to measure teacher quality (16.4% disagreement; per-type teacher agreement: membership 78.0%, equivalence 86.8%) and to study what the student should learn from: the teacher's agreement region (filtered labels) or its full output distribution (raw labels). Because only 7.3% of filtered queries are positive, we train the main student on a class-balanced 2,752-query subsample (1,376 pos/1,376 neg) of the surviving set; a full-corpus (18.7K, natural-prior) variant is reported in §5.2. We ablate filtering while holding balancing fixed with a 2×2 design (label mode × class prior; §5.2).

3.4 Student training

The student is Qwen2.5-1.5B-Instruct with LoRA (r=16, α=32, all attention+MLP projections). Each example is a chat turn: system prompt defining the two query types, user message with the query, assistant answer "Yes"/"No". We apply swap augmentation (each equivalence query appears as (x, y) and (y, x)) to mitigate position bias (Zhu et al., 2025), and supervise only the answer tokens plus the chat-template tail (<|im_end|> \n), located explicitly via answer-token matching (Appendix A). The main model (Tables 1–3) uses 3 epochs, lr 1×10^{-4} cosine, effective batch 32 (per-device 8 × 4 gradient-accumulation steps) on one A40; the 2×2 de-confound models (§5.2) use a longer schedule (5 epochs, lr 2×10^{-4}).

3.5 Selection strategies

At test time the verifier answers queries with p_{yes} from the first answer-token logits. We evaluate: random (baseline), oracle (any-pass upper bound), membership-argmax: one membership query per candidate, pick $\arg\max_x p_{\text{yes}}(x)$ (n queries); dedup-hybrid: equivalence queries cluster candidates, then one membership query per cluster representative; tournament: single-elimination with equivalence queries, membership tiebreaks, following ExPairT-LLM's protocol; adaptive early-exit: membership queries in chunks of 4, stopping as soon as the running best reaches $p_{\text{yes}} \geq 0.95$ — on easy tasks it stops after ∼4–5 queries, on hard tasks it degrades to the full scan.

4 Experimental Setup

Data. HumanEval (Chen et al., 2021), MBPP (augmented (Austin et al., 2021)), and TACO (Li et al., 2023) (fn- and stdin-style subsets). Teacher annotation and verifier training use TACO tasks only; HumanEval and MBPP are used exclusively for selection evaluation, so those results measure cross-dataset transfer. Training tasks are disjoint from evaluation tasks at the task level; a task-level dev split (162 TACO tasks, 1,197 queries) is used for calibration and threshold selection. We verified zero task overlap between all splits.

Models. Teacher: Qwen3-8B. Generator: Qwen2.5-3B (base; 8 samples, temp 0.8). Verifiers: Qwen2.5-0.5B/1.5B-Instruct and Qwen2.5-3B (base) with LoRA.

Metrics. pass@1 of the selected candidate; verifier queries per task (each query is a single student forward pass over the query prompt — no decoding; the teacher query instead requires generating a differentiating input and reasoning over two programs with an 8B model); dev AUROC of p_{yes} against execution ground truth (measured on TACO dev queries). Paired bootstrap 95% CIs are reported for all key deltas (10K resamples over tasks, resampling task indices jointly for the paired difference, seed 7; §5.2); paired binary outcomes are additionally tested with exact McNemar tests. The analysis script (code/significance.py) is released with the project.

Table 1: Selection quality (pass@1) and verifier cost on HumanEval+MBPP (591 tasks, 8 candidates/task). All strategies use the same candidate pools. Queries are 1.5B-verifier forward passes; the adaptive row's per-task query count comes from its own run (5.5) while the membership column of Table 2 reports the full-scan count (8.0) from the same model; ExPairT-LLM's oracle cost is ∼56 frontier-LLM queries per task.

Strategy	Queries/task	HumanEval	MBPP	HE+MBPP	Remaining gap
Random	0	62.8	65.1	64.5	22.5
PairDistilltournament	20.7	64.0	67.2	66.3	20.7
PairDistillmembership-argmax	8.0	63.4	69.3	67.7	19.3
PairDistilladaptive early-exit	5.5	62.2	68.9	67.0	19.9
Oracle (any-pass)	–	89.0	86.2	87.0	–

5 Results

5.1 Main results

Table 1 shows results on HumanEval+MBPP (cross-dataset: the verifier was trained on TACO only). Membership-argmax improves over random by +3.2 (95% CI $[+0.0, +6.4]$, paired bootstrap — borderline) and over tournament by +1.4 (CI $[-0.8, +3.6]$, n.s.) while using the fewest queries; the 1.5B verifier recovers ∼14% of the random–oracle gap on this benchmark. Adaptive early-exit matches membership-argmax within 0.7 points (n.s.) using only 69% of its queries (5.5 vs. 8.0): once a confidently-passing candidate is found, scanning the remaining lower-confidence candidates only risks flipping the argmax in near-tie regions where the verifier is least reliable. On TACO, where oracle headroom is larger, the same verifier's gains over random are statistically significant (Table 2).

Teacher-at-test-time baseline. A natural question is why distill at all, rather than querying the 8B teacher directly at test time. Running the identical membership-argmax protocol with Qwen3-8B (same prompts, greedy, no thinking mode — matching the distillation setup) on a 150-task HumanEval subset yields 65.3% pass@1, statistically indistinguishable from the 1.5B student's 64.7% on the same tasks (paired Δ +0.7 teacher−student, CI $[-7.3, +8.7]$). Extending the teacher baseline to the full 591-task HumanEval+MBPP evaluation, the teacher scores 64.8% — statistically indistinguishable from random selection on the same pools (64.5%; McNemar $p = 0.92$) — while the distilled student scores 67.7% (paired Δ +2.9 over the teacher, 95% CI $[-0.5, +6.3]$; McNemar $+62/-45$, $p = 0.12$): the student numerically exceeds its teacher at selection. The distilled student matches or exceeds its teacher's selection quality on this benchmark while replacing 8B generation with 1.5B forward passes per query; the distillation value proposition is therefore cost amortization at no quality loss — if anything, a small quality gain.

No-distillation baseline. The same 1.5B model without distillation fails completely: vanilla Qwen2.5-1.5B-Instruct, prompted with the identical judge instructions, achieves 12.3% dev accuracy — it collapses to answering "Yes" to almost everything (98.4% of dev queries), so its accuracy merely tracks the 11.7% positive rate — and its membership-argmax selection is below random picking (63.6% vs. 64.5%, paired Δ −0.8, CI $[-4.4, +2.7]$ over random). Argmax over a nearly-constant score surface is degenerate regardless of the collapse direction. The distilled student's +3.2 over random (67.7%) is therefore attributable to the distillation signal itself, not to the base model's latent judgment ability: the same architecture, prompts, and scoring pipeline yield no selection gain until the model is trained on teacher judgments.

Likelihood baseline. A zero-query alternative is to rank candidates by the generator's own confidence: mean per-token log-probability of each candidate under the generator (Qwen2.5-3B, same prompt as generation). This yields 64.8% on HE+MBPP — indistinguishable from random (paired Δ +0.3, CI $[-3.2, +3.7]$) and 2.9 points below the distilled verifier (CI $[-6.3, +0.5]$). Generator self-confidence carries almost no selection signal at this scale, so the verifier's advantage cannot be replicated by reranking with generation likelihoods.

Table 2: Selection quality on TACO with task-level leakage prevention. Clean rows use evaluation tasks disjoint from the verifier's training tasks; the leaked row shows the same verifier on the unfiltered pool, where 95% of evaluation tasks (and all their candidates) were seen during training. The two pools differ in task composition (random: 15.2 vs. 12.7), so the leaked row illustrates the risk of task-level leakage rather than providing a controlled effect size; membership-argmax over random is significant on both clean TACO sets (+2.4, CI $[+0.2, +4.7]$ fn; +1.4, CI $[+0.1, +2.7]$ stdin).

Eval set	Verifier	Random	Membership	Dedup	Tournament	Oracle
TACO-fn (892)	1.5B filtered	12.7	15.1	14.5	14.7	29.5
TACO-fn (892)	1.5B raw	12.7	14.8	14.0	14.2	29.5
TACO-fn (892)	3B base	12.7	16.6	15.4	15.6	29.5
TACO-fn (1999, leaked)	1.5B filtered	15.2	21.4	19.1	19.3	33.0
TACO-stdin (1500)	1.5B filtered	7.5	8.9	8.7	8.9	18.1

5.2 Ablations

Execution filtering vs. raw labels. Training on raw teacher labels yields higher dev accuracy (86.0% vs. 84.0%) but lower AUROC (0.767 vs. 0.855). The two conditions also differ in class prior: the filtered model trains on a class-balanced 50/50 subsample while the raw model trains on the natural 13% positive prior, so part of the AUROC gap may reflect class balance rather than label quality. To separate the two axes we report a 2×2 de-confound (label mode × class prior, identical query budget and hyperparameters) in Table 4: the AUROC advantage of filtering survives within the balanced column, but the filtering contrast on selection is n.s. within both columns (+1.2 balanced, +0.7 natural); the class-prior axis is the larger split. End-to-end, filtering's effect on selection pass@1 is +0.7–1.2 points (n.s.): since selection consumes rankings rather than 0.5-threshold decisions, filtering's contribution is concentrated in ranking quality rather than in the pass@1 of the selected candidate.

Verifier scale. The 0.5B student reaches 84.7% dev accuracy — higher than the 1.5B student's 84.0%, with higher membership accuracy as well (79.3% vs 77.5%) — yet its selection matches random picking exactly, while the 1.5B student gains +3.2 points: pairwise accuracy saturates before selection quality does. Figure 1 makes the puzzle concrete: the two models' p_{yes} distributions over the 1,197 dev queries are nearly indistinguishable (both strongly bimodal; KS statistic 0.481 vs. 0.462, ECE 0.156 vs. 0.161). Pick-level analysis locates the failure instead in within-pool ordering: the 0.5B's argmax agrees with the 1.5B's on only 27.9% of tasks (chance ≈ 12.5%), its picks are spread uniformly over candidate slots (no positional degeneracy), and its pass@1 equals the random expectation to the first decimal — the 0.5B ranks, but by features uncorrelated with passing. A 3B student trained from the base (non-instruct) Qwen2.5 model converges on the verdict token (per-token CE ≈ 0) but its training loss plateaus at 2.59, entirely attributable to the template tail <|im_end|> (CE = 7.78), which the base model never learned to emit. Its selection is the strongest of all verifiers on TACO-fn (16.6%, CI $[+1.7, +6.2]$ over random) and matches 1.5B on HE+MBPP: verdict-token learning, not template-tail likelihood, determines selection quality.

Distillation volume. We subsample the balanced filtered training queries at 25/50/100%. Dev AUROC degrades monotonically with less data (0.855 → 0.843 → 0.811), but task-level selection saturates early: on TACO-fn, 25% of the data already recovers 14.3% (vs. 15.1% full), and all volume differences are within noise (the HE+MBPP curve is non-monotone: 67.2 → 64.5 → 67.7). Selection quality is robust to a 4× reduction in distillation volume even though dev accuracy is not.

Swap augmentation. Training on single-direction equivalence queries only (no $(x, y)/(y, x)$ duplication) leaves selection unchanged: 68.0% membership-argmax on HE+MBPP vs. 67.7% for the main recipe (paired Δ +0.3, CI $[-2.2, +2.9]$, n.s.), with comparable dev accuracy (86.9% vs. 84.0%). The swap augmentation is therefore not load-bearing for se-

Table 3: Ablations. Top: label mode (filtered vs. raw teacher labels). Middle: verifier scale. Bottom: distillation volume (25/50/100% of the balanced filtered subset) and swap augmentation. Selection numbers are pass@1 with membership-argmax; HE+MBPP unless noted.

Ablation	Verifier	Dev acc	Dev AUROC	HE+MBPP	TACO-fn
filtered (main)	1.5B Instruct	84.0	0.855	67.7	15.1
raw	1.5B Instruct	86.0	0.767	67.3	14.8
scale	0.5B Instruct	84.7	0.845	64.5	–
scale	3B base	86.0	0.857	68.4	16.6
volume 25%	1.5B Instruct	77.3	0.811	67.2	14.3
volume 50%	1.5B Instruct	83.6	0.843	64.5	15.4
volume 100%	1.5B Instruct	84.0	0.855	67.7	15.1
no swap aug.	1.5B Instruct	86.9	0.758	68.0	–

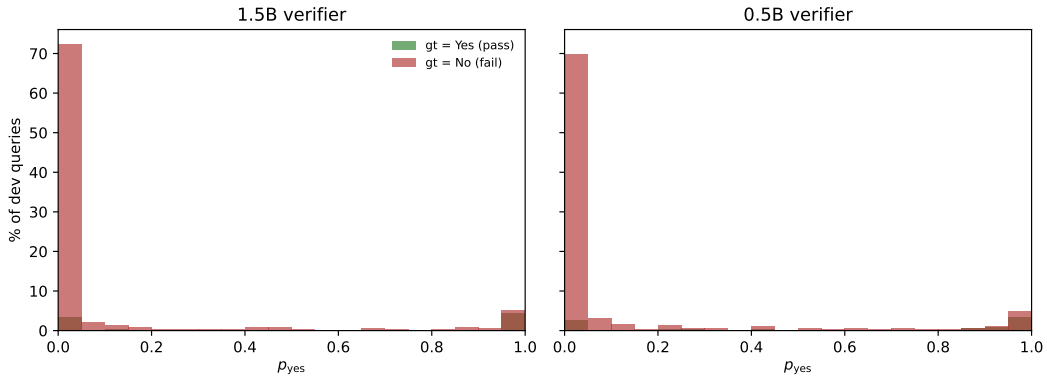


Figure 1: Scored p_{yes} distributions on the 1,197-query dev set. The 0.5B and 1.5B verifiers are nearly indistinguishable at the score-distribution level (both strongly bimodal, similar calibration) — the 0.5B's selection failure lies in within-pool ordering, not in score separation or calibration.

lection quality at this scale, though we retain it in the main recipe as a low-cost safeguard against position bias.

The volume rows subsample the balanced 2,752-query set.

The de-confound refines the mechanism claim. Within the balanced column, filtering's AUROC advantage persists (0.861 vs. 0.768 unified), but its end-to-end selection advantage is +1.2 (McNemar $p = 0.20$, n.s.); within the natural-prior column it is +0.7 ($p = 0.50$). Balancing's effect is positive in both label modes (+1.7 filtered, $p = 0.10$; +1.1 raw, $p = 0.22$; both n.s.). Under the unified schedule, however, the filtered+balanced cell is the only cell significantly above random (15.5, $+62/-37$, McNemar $p = 0.015$) and is significantly above the unified raw+natural cell ($\Delta + 2.35$, $p = 0.011$; the latter survives a Holm correction across the six contrasts, adjusted $p = 0.044$); the remaining four of six pairwise contrasts are n.s. Expressed as net gains over the class-prior-calibrated baseline, balancing the query prior contributes +1.7 (filtered) and +1.1 (raw) points of TACO-fn pass@1, whereas label filtering contributes +1.2 (balanced) and +0.7 (natural) points: the prior axis is the larger of the two and the filtering axis is second-order at this scale, which answers directly what the distilled pairwise oracle adds beyond a class-prior correction. We conclude that the filtering-vs-raw selection gap in Table 3 is explained by the class prior rather than by a filtering mechanism: the label-filtering axis remains second-order at this scale, but the class-prior axis is the largest split in the unified 2×2 (balanced cells 15.5/14.2 vs. natural cells 13.8/13.1), and the headline configuration (filtered + balanced) is the only cell whose selection gain over random is individually significant on TACO-fn.

Table 4: 2×2 de-confound of the execution-filtering claim: label mode (filtered vs. raw) × class prior (balanced 50/50 vs. natural positive), identical query budget (2,752). The original comparison (Table 3, top) varied both factors simultaneously. All four cells were trained under a single unified schedule (5 epochs, lr 2×10^{-4}, effective batch 16). The natural-prior positive rate differs by label mode (7.3% for filtered survivors vs. 13.3% for the raw corpus) because filtering removes teacher errors, most of which are false positives.

Label mode	Class prior	Dev AUROC	TACO-fn pass@1
filtered	balanced (unified)	0.861	15.5
filtered	natural 7.3%	0.802	13.8
raw	balanced	0.768	14.2
raw	natural 13.3% (unified)	0.755	13.1

Seed variance and a scale-up replication. A second seed of the headline configuration (balanced 2,752, seed 13) reproduces dev metrics (84.7% acc, 0.852 AUROC vs. 84.0%/0.855) and selection within noise (67.0% vs. 67.7%, paired Δ −0.7, CI $[-2.5, +1.4]$): the headline result is seed-stable. Pooling the two seeds of this configuration at the observation level (1,182 task×seed pairs) gives $+121/-87$ discordant pairs, McNemar $p = 0.022$ (pooled Δ +2.9): each seed alone is borderline, but the direction is consistent across seeds and the pooled two-seed test is significant. The full-corpus raw-label recipe (natural 13.3% prior) pools significantly as well across its two seeds ($+126/-86$, McNemar $p = 0.007$, pooled Δ +3.4), though its per-seed outcomes are more dispersed (seed 7: 68.5%, +4.1 over random, $p = 0.026$; seed 13: 67.2%, +2.7, n.s.). The filtered-label recipe at its natural 7.3% prior (full 18,757-query corpus) has been run at a single seed (13); that run shows the dev/selection dissociation in its sharpest form: the best dev metrics of any 1.5B model in this paper (89.9% accuracy, 0.876 AUROC) yet selection at random level (64.8%, Δ −2.9 natural-prior − balanced model, CI $[-5.8, 0.0]$). A pick-level analysis localizes the failure to within-pool ordering under a compressed score scale, mirroring the 0.5B signature: the seed-13 model answers "Yes" ($p_{\text{yes}} > 0.5$) to only 7.1% of the 1,197 dev queries (vs. 14.5% for the balanced model), so the expected number of confident "Yes" verdicts in an 8-candidate pool is 0.57 — most pools contain no query above the decision threshold, and argmax operates in a near-zero noise band where the median true positive scores $p_{\text{yes}} = 0.07$ (vs. 0.58 for the balanced model) and ordering is determined by tiny score differences that do not track within-pool correctness. Its pick agreement with the oracle is 17.6% of all 591 tasks (chance 12.5%; balanced model 19.1%), and its global AUROC remains high (0.876) — the failure is within-pool, not global. Because this cell has been run at a single seed, seed variance and label-mode effects cannot be separated there; the raw-label pair's smaller per-seed spread (68.5% vs. 67.2%, seed contrast n.s.) shows that natural-prior training is not uniformly unstable, so we read the filtered-natural run as one draw from a high-variance regime rather than a "failure of scale". Either way it sharpens the paper's central dissociation — query-level metrics do not predict selection quality — and cautions against single-seed comparisons at natural prior.

5.3 Cost analysis

The teacher is queried only at training time (23.7K queries, one-off). At inference, membership-argmax uses 8.0 verifier queries/task (≈4.1K student prefill tokens per task at ∼515 tokens per query, single forward pass, no decoding), versus ExPairT-LLM's ∼56 frontier-model generation queries per task. The saving has two components: a 5.3× parameter ratio (8B teacher vs. 1.5B student) and, more importantly, the generation-vs-forward-pass asymmetry — each teacher query must generate a differentiating input and verdict (hundreds of decoded tokens), while each student query is one prefill over the same prompt. The one-off teacher cost (23.7K queries) amortizes over deployment volume: after ∼3,000 evaluated tasks, the student's cumulative query cost falls below a single re-run of teacher annotation.

6 Discussion and Limitations

When does the verifier help? Gains scale with oracle headroom. On TACO-stdin (oracle 18.1%), membership-argmax improves over random by a relative +18.7% (significant); on TACO-fn (oracle 29.5%) by +18.9% (significant); on HumanEval+MBPP (oracle 87.0%) by +5.0% (borderline). When most pools already contain a passing candidate, selection matters less; distilled verifiers pay off where candidate quality is low and heterogeneous. Notably, the verifier transfers zero-shot from TACO training tasks to HumanEval+MBPP selection while retaining most of its advantage. Limitations. (i) Our verifier inherits the teacher's biases within the agreement region; queries the teacher gets wrong are excluded rather than corrected. (ii) Equivalence queries without a discoverable differentiating input are labeled equivalent, which can be wrong on unprobed inputs (such pairs are a minority of equivalence queries); quantifying sensitivity to this labeling choice is a natural next step. (iii) Gains are bounded by the oracle headroom of the evaluation set: on HumanEval+MBPP, where candidate pools often already contain a passing candidate, that headroom is small and the gains are correspondingly small, while harder distributions (TACO) are where distilled verifiers help most. (iv) All experiments use one generator family (Qwen2.5-3B) and one teacher; cross-generator transfer and stronger-teacher ceilings are natural next steps for extending these findings. (v) Each configuration is a single training run; we report paired bootstrap CIs over tasks. A second seed (13) of the headline configuration reproduces dev metrics (84.7%/0.852 vs. 84.0%/0.855) and selection within noise (67.0% vs. 67.7%, paired Δ -0.7, CI $[-2.5, +1.4]$). (vi) The 2×2 de-confound of execution filtering was run at one query budget (2,752); other budgets may behave differently.

7 Conclusion

PairDistillshows that test-time pairwise LLM oracles for code selection can be amortized into 0.5–1.5B verifiers distilled from a frontier teacher, with membership-argmax selection offering the best accuracy–cost trade-off and adaptive early-exit cutting its cost by a further ∼30% at equal quality. A 2×2 de-confound shows the distillation recipe with a balanced query prior — not execution-filtering of teacher labels — carries the end-to-end result, even though filtering improves ranking metrics. The scale-threshold finding, the ranking-vs-accuracy dissociation, and the volume-robustness of selection quality open questions about what pairwise distillation measures and what selection requires.

AI use statement

Large language models were used throughout this project, and we state this openly rather than in vague terms. Concretely, LLM-based assistants were used for (i) implementing and refactoring the experimental harness and the analysis scripts, (ii) assisting with literature search and the first-pass organisation of the related-work sections, (iii) language polishing and condensation of the manuscript, and (iv) suggesting candidate prompt phrasings and code templates that the authors then reviewed, modified and validated. The pairwise oracle pipelines and the data-analysis scripts that turn judge verdicts into the reported selection accuracies were written with substantial assistant help; the authors verified each reported number against the stored judge outputs. The research question, the design of the method, the evaluation protocol and the interpretation of the results are the authors' own. Every experimental result reported here was produced by actually executing the code on the models described in the paper; no number, figure or reference was generated by a language model and accepted without author verification against the raw run logs or the original publication. The authors take full responsibility for the content of the paper and for any remaining errors.

Ethics statement

The study uses only public code benchmarks (HumanEval, MBPP) and publicly released open-weight models; no human subjects, no private code repositories and no personal data are involved. The distilled verifier inherits the behaviour of the frontier teachers used to

label its training pairs, so any systematic preference of those teachers is transferred to the student; this is a documented limitation of the approach and the reason we evaluate on held-out benchmarks and against oracle judges. We do not recommend deploying the resulting selector as an unattended filter for consequential decisions about code or candidates for review; it is intended as a cheap candidate-ranking component whose errors remain visible at the pass@k level. Because the method trains on model-generated judgments rather than human labels, its agreement with human preference is measured, not assumed. The authors declare no competing interests.

Reproducibility statement

The benchmark splits, the identity and decoding parameters of the teacher models, the number of sampled candidates per problem, the pairwise labelling budget, and the training hyper-parameters of the distilled verifier are all specified in the method section and the appendix; the exact prompts used for the judges appear in the appendix. Every reported selection accuracy is obtained by re-running the released selection pipeline on the stored candidate pools, and the confidence intervals are computed on the same per-problem results, with the number of problems and the resampling procedure stated in the corresponding table captions. Code, the teacher-labeled training queries, the prompt templates and the candidate pools will be released publicly upon acceptance, together with the per-task selection records; these materials are withheld at submission time only because the repository metadata would identify the authors under double-blind review.

References

Miltiadis Allamanis and Pengcheng Yin. Disproving program equivalence with llms. arXiv preprint arXiv:2502.18473, 2025.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. arXiv preprint arXiv:2108.07732, 2021.

Bei Chen, Fengji Zhang, Anh Nguyen, Daoguang Zan, Zeqi Lin, Jian-Guang Lou, and Weizhu Chen. Codet: Code generation with generated tests. In International Conference on Learning Representations (ICLR), 2023.

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

Mouxiang Chen, Zhongxin Liu, He Tao, Yusu Hong, David Lo, Xin Xia, and Jianling Sun. B4: Towards optimal assessment of plausible code solutions with plausible tests. arXiv preprint arXiv:2409.08692, 2024.

David Cho, Yifan Wang, Fanping Sui, and Ananth Grama. Inference-time code selection via symbolic equivalence partitioning. arXiv preprint arXiv:2604.06485, 2026.

Chao-Wei Huang and Yun-Nung Chen. Pairdistill: Pairwise relevance distillation for dense retrieval. In Proceedings of EMNLP, 2024.

Jeevana Priya Inala, Chenglong Wang, Mei Yang, Andres Codas, Mark Encarnación, Shuvendu K. Lahiri, Madanlal Musuvathi, and Jianfeng Gao. Fault-aware neural code rankers. In NeurIPS, 2022.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. Prometheus 2: An open source language model specialized in evaluating other language models. In EMNLP, 2024.

Rongao Li, Jie Fu, Bo-Wen Zhang, Tao Huang, Zhihong Sun, Chen Lyu, Guang Liu, Zhi Jin, and Ge Li. Taco: Topics in algorithmic code generation dataset. arXiv preprint arXiv:2312.14852, 2023.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. Pairjudge rm: Perform best-of-n sampling with knockout tournament. arXiv preprint arXiv:2501.13007, 2025.

Zeyao Ma, Xiaokang Zhang, Jing Zhang, Jifan Yu, Sijia Luo, and Jie Tang. Dynamic scaling of unit tests for code reward modeling. arXiv preprint arXiv:2501.01054, 2025.

Neeva Oza, Ishaan Govil, Parul Gupta, Dinesh Khandelwal, Dinesh Garg, and Parag Singla. Cetbench: A novel dataset constructed via transformations over programs for benchmarking llms for code-equivalence checking. arXiv preprint arXiv:2506.04019, 2025.

Indraneil Paul, Goran Glavaš, and Iryna Gurevych. Themis: Training robust multilingual code reward models for flexible multi-criteria scoring. arXiv preprint arXiv:2605.00754, 2026.

Zhihong Sun, Yao Wan, Jia Li, Hongyu Zhang, Zhi Jin, Ge Li, and Chen Lyu. Sifting through the chaff: On utilizing execution feedback for ranking the generated code candidates. In ASE, 2024.

Hung Quoc To, Minh Huynh Nguyen, and Nghi D. Q. Bui. Functional overlap reranking for neural code generation. In Findings of the Association for Computational Linguistics: ACL 2024, 2024.

Yidong Wang, Zhuohao Yu, Zhengran Zeng, Linyi Yang, Cunxiang Wang, Hao Chen, Chaoya Jiang, Rui Xie, Jindong Wang, Xing Xie, Wei Ye, Shikun Zhang, and Yue Zhang. Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. arXiv preprint arXiv:2306.05087, 2023.

Anjiang Wei, Jiannan Cao, Ran Li, Hongyu Chen, Yuhui Zhang, Ziheng Wang, Yuan Liu, Thiago S. F. X. Teixeira, Diyi Yang, Ke Wang, and Alex Aiken. Equibench: Benchmarking large language models' reasoning about program semantics via equivalence checking. arXiv preprint arXiv:2502.12466, 2025.

Yan Yu, Yilun Liu, Minggui He, Shimin Tao, Weibin Meng, Xinhua Yang, Li Zhang, Hongxia Ma, Dengye Li, Daimeng Wei, Boxing Chen, and Fuliang Li. Elspr: Evaluator llm training data self-purification on non-transitive preferences via tournament graph reconstruction. arXiv preprint arXiv:2505.17691, 2025. AAAI 2026.

Tom Yuviler and Dana Drachsler-Cohen. Expairt-llm: Exact learning for llm code selection by pairwise queries. arXiv preprint arXiv:2511.10855, 2025. AAAI 2026.

Lianghui Zhu, Xinggang Wang, and Xinlong Wang. Judgelm: Fine-tuned large language models are scalable judges. In International Conference on Learning Representations (ICLR), 2025. arXiv:2310.17631.

A Loss-masking alignment

The Qwen chat template appends <|im_end|>\n after assistant content. We locate the answer-token subsequence by exact token matching and supervise from its start to the sequence end. We verified the supervised token set by decoding masked positions before training.

B Reproducibility

Code, data construction scripts, and training configurations are released publicly upon acceptance, together with the prompt templates and the per-task selection records described below. All models are from Qwen2.5/Qwen3 families; evaluation uses the official test suites with the check() invocation fix for HumanEval. All confidence intervals are computed by code/significance.py: paired task-level bootstrap, 10,000 resamples, seed 7, percentile intervals; paired binary outcomes are additionally tested with exact McNemar tests (two-sided exact binomial on discordant pairs). Per-task selection records (results/*.jsonl) are released so every reported delta and CI can be recomputed.