DISTILLTRACE: TRACING LLM DISTILLATION VIA KEYED SOLUTION PREFERENCES

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language model outputs can be collected for unauthorized distillation, motivating providers to determine whether a student was trained on their answers. Yet a watermark detectable in teacher outputs may not survive distillation. Practical tracing therefore requires a learnable signal, broad task coverage, and preserved teacher utility. We introduce DistillTrace, which encodes consistent solution preferences as a watermark. Two LoRA adapters learn response families from quality-filtered, self-distilled answer pairs. Keyed semantic hashing assigns one family to each predicted question category, giving related questions a consistent preference that students can learn through ordinary distillation. A fixed private detector supports black-box tracing and selects watermarked answers from other generators without retraining. Construction and embedding require neither proxy students nor token-level decoding biases. In a matched Qwen-to-Llama comparison on MBPP, aggregating 256 responses yields 99.88% AUROC and 97.77% TPR against matched clean students. Teacher utility is maintained or improved on all five tasks evaluated in this comparison. Separately, question-count-weighted teacher utility rises from 53.73% to 54.50% across four benchmarks. In multi-domain distillation, the semantic-hashing configuration achieves stronger aggregate student detection than the non-semantic configuration. Across four generators and three tasks held out during adapter and detector training, online selection from four candidates achieves 95.52%–100.00% aggregate watermark-detection TPR over 256 responses, extending embedding without additional training.

1 INTRODUCTION

Answers from a deployed large language model (LLM) can be collected and used to train a student model through distillation, often without the provider's consent. This can let third parties reproduce and monetize capabilities derived from a proprietary model while making provenance and license enforcement difficult. Existing defenses either suppress what students can learn (Savani et al., 2026) or embed signals in teacher outputs that can later be traced in students (Sander et al., 2024; Zhao et al., 2025). Suppression may not generalize to an attacker's unknown model or training procedure (Ma et al., 2026), and can also reduce the usefulness or increase the cost of teacher outputs (Savani et al., 2026). We therefore study *tracing*: deciding from black-box queries whether a suspect model was trained on the teacher's answers. Watermarks and fingerprints can identify a model or its outputs (Kirchenbauer et al., 2023; Xu et al., 2024; Jin et al., 2024), but detectability in the teacher does not ensure that the signal is inherited by the student (Gu et al., 2024). The central question is what kind of signal a student reliably inherits without compromising the teacher's utility.

These observations suggest three practical requirements. First, reliable inheritance requires sufficiently strong and consistent signals in the distillation data. Token-level watermarks such as KGW favor a green list defined by hashing preceding tokens, so slight wording changes can alter the partition without changing meaning (Kirchenbauer et al., 2023). Relatedly, Gu et al. (2024) find that longer hash contexts weaken student watermark detection. Semantic watermarks use sentence embeddings to reduce sensitivity to wording (Hou et al., 2024; Liu et al., 2024), but their response-dependent partitions do not ensure consistent response patterns across semantically similar prompts. ADFP instead optimizes teacher responses using proxy-student gradients to improve learnability (Xu et al., 2026), but these gains may not generalize to actual students.

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.

Second, the watermark must cover the domains queried by the distiller, which the provider cannot anticipate. A watermark effective only in certain domains may be too sparsely represented in student training data to support tracing. ReasMark selects watermark tokens from frequent prompts in a target domain (Lv et al., 2026), assuming that the distiller queries that domain.

Third, the watermark should preserve teacher utility. Decoding-time watermarks perturb token sampling for every answer (Kirchenbauer et al., 2023), so stronger detection comes at a quality cost for all users.

A practical watermark for tracing distilled models should therefore be reliably inherited through distillation, remain effective across diverse domains, and preserve teacher utility. We propose DistillTrace (Figure 1), building on the premise that students can inherit a teacher's consistent solution preferences through distillation. DistillTrace uses a private key to assign one of two response families to each semantic question category, encoding the resulting preference as a watermark. The teacher answers with the matching LoRA adapter, so no token-level decoding bias is added.

Constructing response families. We automatically construct paired responses through self-distillation: a model first answers a prompt, then generates an alternative solution conditioned on the prompt and its initial answer. After verification and filtering against ground truth, the two sets of responses are used to train separate LoRA adapters. This procedure applies to domains with ground-truth supervision where valid alternatives can be generated, without requiring existing two-solution datasets, domain-specific rewriting rules, or proxy-student optimization. We embed the preference in final answers because reasoning traces may be withheld by the provider or discarded by the distiller; OpenAI's GPT-5-series reasoning models and o4-mini, for example, expose only reasoning summaries.[1] A signal confined to hidden reasoning never reaches the student.

Keyed semantic hashing. A question classifier maps each question to a semantic category, and the private key assigns a response family to that category, selecting the corresponding adapter. Related questions within a category therefore reinforce the same preference despite differences in wording. This provides a stable association that students can learn through ordinary distillation without knowing the key or reproducing the hashing rule.

Detection and tracing. A private response detector is trained on the self-distilled pairs to distinguish the two families and then frozen. During verification, the provider scores student responses against the keyed family assignments and averages the scores across queries. Tracing requires only black-box access to text responses, without student parameters or training records. Students receive only ordinary question–answer pairs; the key and detector remain private.

Online embedding. For unseen domains or models without adapters, we reuse the same detector to select among model-generated candidate answers according to the keyed family assignment. This extends watermark embedding beyond the training domains and to other generators without new adapters or detector retraining.

In same- and cross-family MBPP distillation, aggregating 256 responses yields true-positive rates (TPRs) of 95.33–98.89% for keyed students versus 2.87–32.04% for random-selection controls (Table 4). Across four teacher benchmarks, DistillTrace keeps weighted utility at 54.50% (Base: 53.73%) with 82.28% TPR at a false-positive rate (FPR) of at most 1%, whereas KGW reaches only 71.46% TPR at $\delta = 3$, where utility drops to 44.65% (Table 1). In a matched comparison, DistillTrace is the only method whose teacher utility does not fall below Base (Table 5). In summary, our contributions are threefold:

- **Keyed solution preferences as an inheritable watermark.** We show that a student inherits which valid solution a teacher prefers when that preference is keyed to the semantic bucket of each question. Related questions repeat the same preference, so the signal is learned through ordinary distillation without proxy students or decoding-time token bias, and teacher utility is preserved.
- **Extensible embedding and verification.** Self-distillation builds answer pairs from ordinary prompts without pre-paired solutions; the same fixed detector both verifies suspect models and selects marked answers from unadapted generators on held-out tasks, covering new domains and models without retraining.

[1] See the OpenAI documentation on reasoning models.

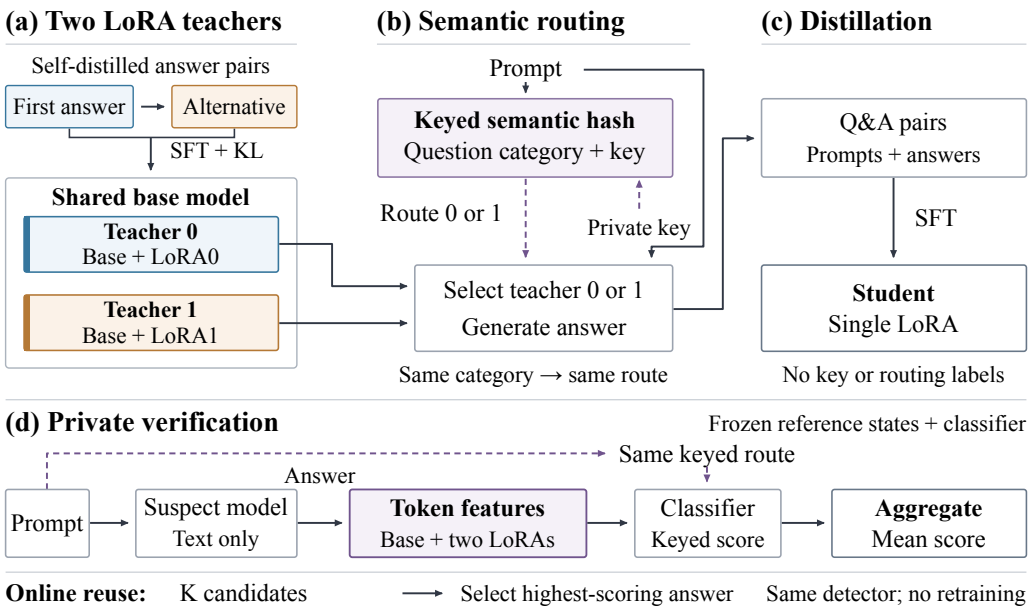


Figure 1: Overview of DistillTrace. (a) Self-distilled answer pairs train two LoRA teachers on a shared base. (b) Keyed semantic hashing assigns each question category a private bit $b \in \{0, 1\}$, selecting teacher b. (c) Students learn from question–answer pairs without keys or routing labels. (d) A detector uses frozen reference models to score responses, aggregates scores for black-box tracing, and selects candidates for online embedding. Colors distinguish the response families; dashed links indicate key-dependent control.

- **Evaluation of tracing and utility.** We evaluate teacher utility, online selection, and same- and cross-family inheritance against KGW-based tracing methods (Tables 1–5).

2 Related Work

We first review methods for verifying ownership of a model or its outputs (Tramèr et al., 2016; Orekondy et al., 2019; Jagielski et al., 2020), then defenses against unauthorized distillation through suppression or student tracing. Table 10 in Appendix C compares designs and access requirements.

Watermarking. Watermarking embeds a verifiable signal in generated outputs or model behavior (Adi et al., 2018; Jia et al., 2021; He et al., 2022; Zhao et al., 2023b; Kirchenbauer et al., 2024; Jovanović et al., 2024). Inference-time methods such as KGW bias token sampling without changing model weights (Kirchenbauer et al., 2023; Hu et al., 2024; Zhao et al., 2023a; Christ et al., 2024; Wu et al., 2023). Model-level ownership marks can be implanted through training: Xu et al. (2024) use instruction tuning to implant a private-key trigger-response fingerprint and verify that it persists after fine-tuning the marked model. This result does not establish that a student trained on the model's outputs will inherit the trigger. Distortion-free sampling, SemStamp, in-context watermarking, and ReasonMark provide other output-level signals (Kuditipudi et al., 2023; Hou et al., 2024; Liu et al., 2026b;a). Methods that bias token sampling trade text quality for detectability (Kirchenbauer et al., 2023); in our evaluation, raising KGW's δ from 1 to 5 lowers teacher utility from 52.60% to 20.20% (Table 1).

Fingerprinting. Passive fingerprinting identifies characteristic responses without changing the protected model or its generation; ProFLingo is a black-box example (Jin et al., 2024; Gloaguen et al., 2026). It avoids embedding-induced utility loss, but detection can weaken if adaptation changes those responses (Yang et al., 2026b). The signal can be lost during distillation (Gu et al., 2024; Sander et al., 2024; Pan et al., 2025), leaving the student untraceable by these methods.

Suppressing distillation. Suppressing distillation aims to limit how much an unauthorized student learns from a teacher's outputs. ADS steers token sampling using proxy-student gradients (Savani et al., 2026); DOGe and Fang et al.'s logit transformation also optimize against proxy students (Li et al., 2025; Fang et al., 2026). Ma et al. use proxy students to optimize instructions for rewriting teacher reasoning traces (Ma et al., 2026). ADS and DOGe report transfer to the students they evaluate (Savani et al., 2026; Li et al., 2025), but because these methods optimize against chosen proxies, their effectiveness against an unseen distiller with a different architecture or training recipe is not guaranteed. Even if suppression transfers, it cannot establish whether a suspect student was trained on the protected teacher's outputs. DistillTrace instead tests student outputs for an inherited signal without proxy-student optimization.

Tracing distillation. *CoT-based tracing.* CoT offers an additional watermarking channel without directly modifying final answers: the watermarking branch of trace rewriting inserts trigger–target associations into teacher reasoning traces (Ma et al., 2026; Liu et al., 2025). ReasMark ties private prompt-token partitions to reasoning behavior and detects the inherited pattern in a distilled student (Lv et al., 2026). These approaches rely on marked reasoning traces entering the training data and being available for verification; this channel is often unavailable. Reasoning APIs may return only summaries of the model's reasoning (Section 1), and students can be distilled from answer text alone, as in prior tracing work (Sander et al., 2024; Gu et al., 2024); ReasMark's verification further requires the suspect model to expose its reasoning (Lv et al., 2026). DistillTrace instead marks the final answer, which every distiller must train on, so tracing works whether or not reasoning traces are kept or exposed.

KGW-based tracing. KGW selects keyed green tokens at each decoding step, boosts their probability, and detects their excess in generated text (Kirchenbauer et al., 2023). Radioactivity shows that a student fine-tuned on KGW-watermarked outputs can retain a weak token-level bias and provides tests with valid p-values under open- and closed-model access (Sander et al., 2024). LIDet strengthens black-box detection with anchor models that select informative queries and adapt thresholds across hash keys (Zhao et al., 2025). ADFP uses a proxy student to guide token sampling toward choices that raise the student's green-token rate (Xu et al., 2026), but it still perturbs decoding and relies on transfer from the proxy; in our matched comparison, its teacher utility falls 3.80–10.20 pp below Base (Table 5). DistillTrace adds no token-level bias: its students keep higher utility (53.61%) than students under every tested KGW setting (49.22–52.52%; Table 3), and its teacher utility exceeds Base (Table 1).

Transformation-based tracing. CodeGenGuard trains a code LM to apply semantics-preserving transformations under a private trigger and uses shadow training so that the pattern survives model extraction; verification compares transformation rates with and without the trigger (Yang et al., 2026a). It is evaluated only on code generation and requires modifying the protected model. DistillTrace covers mathematics, programming, instruction following, and science, and its fixed detector also marks outputs of unadapted generators online (Tables 2 and 4).

3 METHOD

DistillTrace links the semantic group of a question to a private response preference. We construct two sets of answers by self-distillation, train two LoRA adapters and a private detector on these answers, and use a keyed semantic hash to choose the preferred set for each question. The same private detector supports teacher watermark detection, online answer selection, and testing whether a student inherits the preference (Figure 1).

3.1 OVERVIEW AND SETTING

Let B be a base language model and T_0, T_1 its two LoRA-adapted teachers. The provider retains a secret key k, a question-to-bucket map g, and a private reference-model detector r. The map g reads the domain and question text; the detector r takes the question, answer, and keyed route and computes features with three frozen reference states: B, T_0, and T_1. The key assigns each predicted bucket a bit $b \in \{0, 1\}$, which selects a response family. A distiller receives only the served question-answer pairs and does not know the key, response-family labels, or adapter identities.

3.2 SELF-DISTILLATION AND TWO LORA ADAPTERS

Constructing answer pairs. For each training question x, a generation model G first produces an answer $y^{(0)}$. We then give G the same question and its first answer, asking for another solution:

$$y^{(0)} \sim G(\cdot \mid x), \qquad y^{(1)} \sim G(\cdot \mid c(x, y^{(0)})). \tag{1}$$

Here c contains the question, the first answer, and the instruction to generate another answer. Labels 0 and 1 record these two generation stages. They define two response families without prescribing the same pair of algorithms or reasoning strategies for every task.

Ground truth or task checks filter generated pairs for quality. A dataset can therefore support preference construction without supplying two solutions per question, provided valid alternatives can be generated and checked. The retained pairs form $\mathcal{D}_b = \{(x_i, y_i^{(b)})\}_{i=1}^{N}$ for $b \in \{0, 1\}$.

Training the adapters. We train one LoRA adapter on each response family, using a shared base model (Hu et al., 2021). Adaptation combines supervised fine-tuning with KL regularization toward B. Both adapters receive the original question x as input. The first answer is used to construct the second target, but is not an input to T_1 during fine-tuning or deployment. The two adapters learn different response preferences while task utility is measured separately.

3.3 PRIVATE REFERENCE-MODEL DETECTOR

Given a prompt x, a response y, and its keyed route $b = b_k(x)$, the private detector teacher-forces y under the three frozen reference states B, T_0, and T_1. For each response token and reference state, it records the observed token's log probability and rank, the top-1 token, and that token's log probability. The two LoRA states are oriented as keyed and opposite according to b. The token observations are summarized in a fixed-length feature vector $\phi(x, y, b)$ of within-state choice statistics, comparisons with B, aggregate likelihood statistics, and route-sign interactions (Appendix A.3).

Let z denote feature standardization fitted on detector training data. The logistic-regression logit is

$$r_b(x, y) = \mathbf{w}^\top z(\phi(x, y, b)) + c. \tag{2}$$

An L2-regularized logistic regression learns $\mathbf{w}$ and c from paired Base (label 0) and keyed watermarked (label 1) responses to questions from the adapter training domains. The reference states remain frozen. Because the keyed route is already encoded in ϕ, the detector output is not multiplied by another route sign. This detector is fitted once and reused without retraining for the DistillTrace results in Tables 1–4, including online answer selection and student tracing. Its logit is a detection score, not a calibrated probability of provenance.

3.4 WATERMARK EMBEDDING

Keyed semantic preferences. Our semantic hash first predicts a bucket from the question with a lightweight per-domain classifier trained on fixed weak-label rules. This is supervised bucket prediction, rather than sentence-embedding clustering. At inference, it reads only the domain and question text. Appendix A.2 gives the bucket rules and fitting settings.

Write $g(x)$ for the predicted bucket, with the domain implicit in x. The private bit is

$$b_k(x) = H_k(g(x)) \in \{0, 1\}, \tag{3}$$

where H_k applies a cryptographic hash function (e.g., SHA-256) to the key and the bucket name and keeps one bit of the digest (Appendix A.2). All questions assigned to the same bucket share a bit. This repeats the same preference across related questions, giving a student a consistent relationship to learn from the answers. The student learns this association without recovering the key; its stability depends on bucket prediction.

Adapter routing. The provider answers a question with the teacher selected by its bit:

$$y \sim T_{b_k(x)}(\cdot \mid x). \tag{4}$$

For offline distillation data, the same bit selects between the two generated teacher answers. The exported data contain only the question and selected answer. Missing teacher answers are not replaced with ground truth. Embedding uses no proxy-student gradients or simulated student updates.

Online selection. The fixed private detector can also select a response from an unadapted generator. Given K candidate answers $\mathcal{C}_K(x)$, it returns

$$y^* = \arg\max_{y \in \mathcal{C}_K(x)} r_{b_k(x)}(x, y). \tag{5}$$

This route can express the private preference on tasks beyond adapter training without training new adapters or a new detector. More candidates require more generation and scoring, and maximizing the route-conditioned detector score can change task utility. We therefore report both utility and detection for each candidate budget.

3.5 Detection and student tracing

For each question x_i and answer y_i from M, the verifier recomputes the private bit and aligns the answer score:

$$s_i = a_k(x_i, y_i) = r_{b_k(x_i)}(x_i, y_i). \tag{6}$$

For m answers, it takes the arithmetic mean,

$$S_m = \frac{1}{m} \sum_{i=1}^{m} s_i, \qquad V_k(M) = S_m. \tag{7}$$

A threshold for this statistic must be calibrated with non-inheriting controls under the same query and aggregation protocol. A single high score is insufficient for attribution: the evidence is agreement with the private bucket-to-bit assignment across answers. Correlated questions and repeated responses affect the null distribution and must be reflected in calibration.

Verification needs only the suspect model's generated answers; token likelihoods are computed under the provider's frozen reference states, not the suspect's. The test applies to served teacher responses and to students trained on them (Appendix A.4), with base models and non-keyed students as controls.

4 Experiments

We evaluate teacher-side watermarking through adapter-based generation and online selection on held-out tasks, then study multi-domain Qwen-to-Qwen distillation under semantic hashing and same- and cross-family MBPP transfer to students. Each experiment pairs detection with task utility. Table 10 in Appendix C compares method designs; the experiments provide measured comparisons.

4.1 Evaluation settings and metrics

Teacher outputs. For adapter-routed generation (Table 1), the keyed bit routes Qwen3.5-9B to one of its two LoRA adapters. We compare with Base and KGW (Kirchenbauer et al., 2023) at $\delta \in \{0.5, 1, 2, 3, 5\}$ on Math500 (Lightman et al., 2024), APPS test (Hendrycks et al., 2021), IFEval (Zhou et al., 2023), and GPQA Diamond (Rein et al., 2023) (500, 5,000, 541, and 198 examples), covering mathematics, programming, instruction following, and science. We sweep KGW because it is the signal underlying Radioactive, LIDet, and ADFP; Table 5 compares these tracing methods directly. CoT-based methods do not apply in this setting: they mark reasoning traces, which are absent from answer-only teacher outputs and are often hidden by commercial reasoning APIs (Section 2). For online selection (Table 2), Qwen3.5-9B, Llama 3.1 8B Instruct, GPT-5.6 Luna, and Claude Haiku 4.5 each generate K candidates without adapters, and the fixed detector selects one. HumanEval (Chen et al., 2021), MBPP (Austin et al., 2021), and Spider dev (Yu et al., 2018) are held out from adapter and detector training, though not necessarily from the generators' pretraining.

Distilled students. Table 3 distills hash-routed Qwen3.5-9B teacher answers into a single-LoRA Qwen student across the four teacher domains, with clean distillation, non-semantic hashing, and KGW as controls. Table 4 tests same- and cross-family transfer on MBPP: unadapted Qwen3.5-9B or Llama 3.1 8B Instruct teachers generate candidates for each training question, the fixed detector selects a keyed answer, and a single-LoRA student is trained on the selected pairs; random-selection students trained on the same questions serve as controls. Students answer test questions directly, without candidate selection.

Table 1: Adapter-routed teacher utility and detection (Qwen3.5-9B). Overall is dataset-size weighted, with change from Base (pp) in parentheses; TPR at FPR $\leq 1\%$. Full results in Table 8.

	Task utility (%)					Overall detection (%)	
Method	**Math500**	**APPS**	**IFEval**	**GPQA**	**Overall**	**TPR**	**AUROC**
Base	96.00	46.34	83.73	51.52	53.73 (0.00)	—	—
KGW $\delta = 0.5$	96.00	45.52	85.40	51.52	53.21 (-0.51)	15.16	74.29
KGW $\delta = 1$	**97.00**	44.54	85.03	**55.56**	52.60 (-1.12)	38.17	83.88
KGW $\delta = 2$	95.20	41.58	**86.14**	48.99	49.98 (-3.75)	60.54	91.02
KGW $\delta = 3$	94.40	35.44	85.03	41.41	44.65 (-9.07)	71.46	94.61
KGW $\delta = 5$	56.40	11.26	73.57	8.59	20.20 (-33.53)	**89.36**	98.27
DistillTrace	96.60	**46.90**	85.40	**55.56**	**54.50** ($+0.77$)	82.28	**98.31**

Metrics. Utility is the task score (%), and differences are in percentage points (pp). TPR is reported at FPR $\leq 1\%$. The MBPP student comparison reports AUROC and TPR of keyed and random students against the undistilled Base, using 256 responses per group. Aggregate detection averages aligned response scores. Teacher weighted metrics use $w_d = N_d/6{,}239$ (APPS contributes 80.14%), and weighted AUROC averages per-dataset AUROCs.

Implementation details. Teacher adapters use LoRA rank $r = 16$, $\alpha = 32$, and dropout 0.05, and are trained for four epochs with a learning rate of 2×10^{-5} and an effective batch size of 32, each on four NVIDIA H200 GPUs. The detector's reference states are the Qwen3.5-9B base model and its two teacher adapters. Students use $r = 32$, $\alpha = 64$, dropout 0.05, an effective batch size of 16, and a learning rate of 2×10^{-5}; training budgets are given in Appendix A.4.

4.2 Teacher utility and detection

Table 1 compares task utility, weighted TPR, and weighted AUROC for adapter-routed teacher responses across four benchmarks. DistillTrace reaches weighted utility 54.50%, compared with 53.73% for Base, a gain of 0.77 pp. Its weighted TPR is 82.28%, and weighted AUROC is 98.31%.

Increasing KGW strength generally improves detection while reducing utility. At $\delta = 3$, weighted TPR is 71.46% and utility is 44.65%; at $\delta = 5$, TPR reaches 89.36% but utility falls to 20.20%. DistillTrace therefore improves both metrics over KGW at $\delta = 3$, while KGW at $\delta = 5$ has higher TPR and lower utility.

The per-task results in Table 8 show variation hidden by the weighted mean. DistillTrace improves utility over Base on all four tasks, but Math500 TPR is only 41.88% despite 96.60% utility. KGW at $\delta = 1$ reaches 97.00% utility on Math500, and stronger KGW settings have higher TPR there. These results support a favorable overall trade-off, with task-dependent detection. The four-domain teacher comparison evaluates teacher responses; student inheritance is evaluated separately below.

4.3 Teacher-side online selection on held-out tasks

Table 2 reports online selection at $K = 2$ and $K = 4$, with $K = 1$ as the utility baseline; unlike in Table 1, the detector here chooses among candidates from unadapted generators. Table 9 gives the full Qwen/Llama sweep up to $K = 16$. Llama uses greedy decoding at $K = 1$ and sampled candidates otherwise, and is evaluated on Spider 1.0 dev.

Across the 12 generator–task pairs, aggregate TPR is 55.27–100.00% at $K = 2$ and 95.52–100.00% at $K = 4$, where all HumanEval and MBPP pairs reach 100.00% and GPT-5.6 Luna and Claude Haiku 4.5 reach 99.98% and 95.52% on Spider dev. At $K = 4$, Qwen, GPT, and Claude gain utility on MBPP but lose it on Spider dev, while Llama loses utility on all three tasks.

Larger budgets do not consistently improve utility: Qwen gains 0.61 pp on HumanEval at $K = 16$ (89.02%) but falls to 68.18% on Spider, and every Llama setting with $K \geq 4$ is below its baseline. Because selection maximizes the detector score later used for detection, this TPR measures the selected answers' agreement with the keyed preference and is not directly comparable with Table 1; student inheritance is tested separately below.

Table 2: Online selection on held-out tasks with K candidates per question. U: utility (%); TPR (%) over 256 responses at FPR $\leq 1\%$. Full sweep in Table 9.

		$K = 1$	$K = 2$		$K = 4$	
Dataset	**Generator**	**U**	**U**	**TPR**	**U**	**TPR**
HumanEval	Qwen3.5-9B	88.41	87.20	100.00	84.76	100.00
	Llama 3.1 8B	68.90	68.90	100.00	65.85	100.00
	GPT-5.6 Luna	96.34	97.56	81.44	96.34	100.00
	Claude Haiku 4.5	93.29	94.51	99.01	94.51	100.00
MBPP	Qwen3.5-9B	69.80	71.00	100.00	72.40	100.00
	Llama 3.1 8B	58.80	56.20	96.00	55.20	100.00
	GPT-5.6 Luna	93.20	94.60	97.08	95.00	100.00
	Claude Haiku 4.5	81.80	82.20	91.94	82.20	100.00
Spider dev	Qwen3.5-9B	71.47	71.18	100.00	69.92	100.00
	Llama 3.1 8B	69.34	69.73	84.44	66.83	100.00
	GPT-5.6 Luna	75.53	75.63	65.51	74.47	99.98
	Claude Haiku 4.5	74.37	74.18	55.27	73.98	95.52

Table 3: Multi-domain Qwen3.5-9B distillation: semantic hashing versus controls. TPR at FPR $\leq 1\%$ for single responses and 256-response aggregates.

		Single response		**Aggregate (256)**	
Method	**Utility (%)**	**AUROC (%)**	**TPR (%)**	**AUROC (%)**	**TPR (%)**
Base	53.42	—	—	—	—
Clean distillation	52.51	50.61	1.44	48.69	19.54
Non-semantic hash	52.57	51.76	0.22	59.94	0.18
KGW $\delta = 0.5$	52.52	50.85	1.34	63.68	9.42
KGW $\delta = 1$	52.49	51.88	1.20	70.79	18.79
KGW $\delta = 2$	51.35	54.86	1.08	96.27	65.39
KGW $\delta = 3$	49.22	56.38	1.43	**98.32**	**83.01**
Semantic hash	**53.61**	**70.03**	**1.81**	98.26	78.18

4.4 Multi-domain distillation and semantic hashing

Table 3 compares semantic hashing with non-semantic hashing and other controls in multi-domain Qwen-to-Qwen distillation. Semantic hashing reaches 70.03% single-response AUROC, 78.18% aggregate TPR, and 53.61% utility, whereas non-semantic hashing reaches 0.18% aggregate TPR and 52.57% utility, consistent with students inheriting preferences repeated within semantic groups. Because training set sizes vary across configurations, the table compares complete pipelines rather than isolating hashing alone.

KGW aggregate TPR ranges from 9.42% to 83.01%; at $\delta = 3$ it exceeds semantic hashing in TPR but has lower utility (49.22% versus 53.61%). Clean distillation already reaches 19.54% aggregate detection, so ordinary distillation is a necessary control, and thresholds and negative models must be matched before reading such rates as false positive rates.

4.5 Same- and cross-family watermark inheritance

Table 4 compares four MBPP teacher-to-student pairings, each with a keyed student, a random-selection control, and an undistilled student Base. Same-family keyed and random students match (71.20% for Qwen, 59.80% for Llama). Across families, keyed utility rises by 0.60 pp for Qwen-to-Llama and falls by 8.00 pp for Llama-to-Qwen relative to the student Base.

With 256 responses per group, keyed students reach 99.74–99.87% AUROC and 95.33–98.89% TPR, including 97.21% (Llama-to-Qwen) and 98.89% (Qwen-to-Llama) across families, while random controls reach 2.87–32.04% TPR. Because controls also yield positive detections, keyed-

Table 4: Same- and cross-family MBPP distillation. Gain: keyed minus student Base (pp). Detection is against the undistilled Base over 256 responses; all values in %.

	Utility (%)					Keyed student		Random control	
Teacher → student	**Teacher**	**Student Base**	**Keyed**	**Random**	**Gain**	**AUROC**	**TPR**	**AUROC**	**TPR**
Llama → Qwen	58.60	70.60	62.60	63.20	−8.00	**99.79**	**97.21**	**67.19**	**19.63**
Qwen → Llama	70.60	58.60	59.20	58.80	+0.60	**99.87**	**98.89**	**63.73**	**32.04**
Llama → Llama	58.60	58.60	59.80	59.80	+1.20	**99.74**	**95.33**	**57.56**	**2.87**
Qwen → Qwen	70.60	70.60	71.20	71.20	+0.60	**99.80**	**96.74**	**69.30**	**13.91**

Table 5: Matched Qwen-to-Llama comparison: teacher utility (%; change from Base in pp) and student AUROC/TPR (%) on MBPP vs. matched clean students (256 questions). ADFP: same-backend rerun.

	Teacher utility (%)					Student (MBPP)	
Method	**MBPP**	**Math500**	**HumanEval**	**Spider dev**	**APPS**	**AUROC**	**TPR**
Base	68.00	60.20	87.20	71.57	46.14	—	—
DistillTrace	**69.40 (+1.40)**	**60.40 (+0.20)**	**87.20 (+0.00)**	**71.66 (+0.09)**	**46.16 (+0.02)**	99.88	97.77
ADFP	57.80 (−10.20)	56.40 (−3.80)	80.49 (−6.71)	64.99 (−6.58)	29.54 (−16.60)	99.40	84.86
Radioactive	62.40 (−5.60)	54.20 (−6.00)	84.15 (−3.05)	67.41 (−4.16)	34.08 (−12.06)	**100.00**	**100.00**
LIDet	66.00 (−2.00)	58.40 (−1.80)	85.37 (−1.83)	67.31 (−4.26)	42.32 (−3.82)	99.92	98.18

student scores should be compared with the corresponding non-keyed student rather than read as attribution on their own.

Table 5 compares DistillTrace with ADFP (Xu et al., 2026), Radioactive (Sander et al., 2024), and LIDet (Zhao et al., 2025) in a matched Qwen-to-Llama setting, reporting teacher utility against a shared Qwen Base and student detection against matched clean students.

4.6 Scope of the evidence

Table 3 tests multi-domain inheritance within Qwen using hash-routed teacher LoRAs; Table 4 tests MBPP inheritance across Qwen and Llama after online selection constructs the training answers. Table 5 adds a matched Qwen-to-Llama comparison with prior methods under a shared-Base protocol. HumanEval and Spider dev assess teacher-side online selection. The experiments do not test paraphrasing attacks, mixed-source training, further fine-tuning, or repeated distillation. Multiple seeds and keys, uncertainty estimates, and component ablations remain needed to assess stability and separate preference learning from style differences and gains due to quality-filtered training.

5 Conclusion

We introduced DistillTrace for watermarking teacher outputs and tracing student training. Self-distillation builds paired answers using available ground truth for filtering, without requiring two supplied solutions per question. Two LoRA adapters learn response preferences, keyed semantic hashing repeats them across related questions, and a private detector measures their alignment with the key. The same detector also supports online watermarking on held-out tasks without further training. Across four teacher benchmarks, DistillTrace preserves utility. In multi-domain Qwen-to-Qwen distillation, semantic hashing strengthens student detection compared with non-semantic hashing. In exploratory MBPP distillation, same-family keyed students match their random-selection controls in utility, while Llama-to-Qwen transfer lowers utility. Aggregation strengthens keyed-student detection across same- and cross-family pairings, with nonzero detections among random controls. These results show how useful response preferences can carry evidence of their source through distillation.

AI USE STATEMENT

We used generative AI tools to assist with writing research code and polishing the manuscript. We take responsibility for the final code, analyses, and text.

ETHICS STATEMENT

DistillTrace is intended to trace the use of protected model outputs in student training. A detector score is statistical evidence, not proof of unauthorized use: non-keyed controls in our experiments also yield positive detections. Attribution claims should therefore use matched controls and human review. The secret key and private detector should be accessible only to authorized evaluators. Our experiments query commercial models (GPT-5.6 Luna and Claude Haiku 4.5) through their APIs only to evaluate teacher-side online selection on public benchmarks; no student model is trained on their outputs.

REPRODUCIBILITY STATEMENT

The Method section describes answer-pair construction, LoRA training, semantic routing, and the private detector. The Experiments section specifies the models, benchmarks, metrics, and controls. Appendix A gives implementation and student-training details, and Appendix B reports the full experimental results. We plan to make the code publicly available upon publication.

REFERENCES

Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th USENIX security symposium (USENIX Security 18)*, pp. 1615–1631, 2018.

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

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Miranda Christ, Sam Gunn, and Or Zamir. Undetectable watermarks for language models. In *The Thirty Seventh Annual Conference on Learning Theory*, pp. 1125–1139. PMLR, 2024.

Hao Fang, Tianyi Zhang, Tianqu Zhuang, Jiawei Kong, Kuofeng Gao, Bin Chen, Leqi Zheng, Shu-Tao Xia, and Ke Xu. Towards distillation-resistant large language models: An information-theoretic perspective. *arXiv preprint arXiv:2602.03396*, 2026.

Thibaud Gloaguen, Robin Staab, Nikola Jovanović, and Martin Vechev. Llm fingerprinting via semantically conditioned watermarks. In *International Conference on Learning Representations*, volume 2026, pp. 69713–69748, 2026.

Chenchen Gu, XIANG LI, Percy Liang, and Tatsunori Hashimoto. On the learnability of watermarks for language models. In *International Conference on Learning Representations*, volume 2024, pp. 38745–38768, 2024.

Xuanli He, Qiongkai Xu, Yi Zeng, Lingjuan Lyu, Fangzhao Wu, Jiwei Li, and Ruoxi Jia. Cater: Intellectual property protection on text generation apis via conditional watermarks. *Advances in Neural Information Processing Systems*, 35:5431–5445, 2022.

Dan Hendrycks, Steven Basart, Saurav Kadavath, Mantas Mazeika, Akul Arora, Ethan Guo, Collin Burns, Samir Puranik, Horace He, Dawn Song, et al. Measuring coding challenge competence with apps. *arXiv preprint arXiv:2105.09938*, 2021.

Abe Hou, Jingyu Zhang, Tianxing He, Yichen Wang, Yung-Sung Chuang, Hongwei Wang, Lingfeng Shen, Benjamin Van Durme, Daniel Khashabi, and Yulia Tsvetkov. Semstamp: A semantic watermark with paraphrastic robustness for text generation. 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. 4067–4082, 2024.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. Unbiased watermark for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 45408–45436, 2024.

Matthew Jagielski, Nicholas Carlini, David Berthelot, Alex Kurakin, and Nicolas Papernot. High accuracy and high fidelity extraction of neural networks. In *29th USENIX security symposium (USENIX Security 20)*, pp. 1345–1362, 2020.

Hengrui Jia, Christopher A Choquette-Choo, Varun Chandrasekaran, and Nicolas Papernot. Entangled watermarks as a defense against model extraction. In *30th USENIX security symposium (USENIX Security 21)*, pp. 1937–1954, 2021.

Heng Jin, Chaoyu Zhang, Shanghao Shi, Wenjing Lou, and Y Thomas Hou. Proflingo: A fingerprinting-based intellectual property protection scheme for large language models. In *2024 IEEE Conference on Communications and Network Security (CNS)*, pp. 1–9. IEEE, 2024.

Nikola Jovanović, Robin Staab, and Martin Vechev. Watermark stealing in large language models. *arXiv preprint arXiv:2402.19361*, 2024.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A watermark for large language models. In *International conference on machine learning*, pp. 17061–17084. PMLR, 2023.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. On the reliability of watermarks for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 49660–49704, 2024.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*, 2023.

Pingzhi Li, Zhen Tan, Mohan Zhang, Huaizhi Qu, Huan Liu, and Tianlong Chen. Doge: Defensive output generation for llm protection against knowledge distillation. *arXiv preprint arXiv:2505.19504*, 2025.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, volume 2024, pp. 39578–39601, 2024.

Aiwei Liu, Leyi Pan, Xuming Hu, Shiao Meng, and Lijie Wen. A semantic invariant robust watermark for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 6499–6519, 2024.

Kaiyuan Liu, Shaotian Yan, Rui Miao, Bing Wang, Chen Shen, Jun Zhang, and Jieping Ye. Where did this sentence come from? tracing provenance in llm reasoning distillation. *arXiv preprint arXiv:2512.20908*, 2025.

Shuliang Liu, Xingyu Li, Hongyi Liu, Dong Fang, Bingchen Duan, Zheng Qi, Lingfeng Su, and Xuming Hu. Distilling the thought, watermarking the answer: A principle semantic guided watermark for reasoning large language models. In *International Conference on Learning Representations*, volume 2026, pp. 140799–140829, 2026a.

Yepeng Liu, Xuandong Zhao, Christopher Kruegel, Dawn Song, and Yuheng Bu. In-context watermarks for large language models. In *International Conference on Learning Representations*, volume 2026, pp. 53136–53162, 2026b.

Peizhuo Lv, Ruihua Zhou, Yunpeng Li, Ruigang Liang, Xingshuo Han, XiaoFeng Wang, Wei Dong, and Yuling Liu. Reasmark: A robust watermark for attributing llm reasoning under knowledge distillation attacks. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 47221–47241, 2026.

Xinhang Ma, William Yeoh, Ning Zhang, and Yevgeniy Vorobeychik. Protecting language models against unauthorized distillation through trace rewriting. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11307–11324, 2026.

Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4949–4958. IEEE, 2019.

Leyi Pan, Aiwei Liu, Shiyu Huang, Yijian Lu, Xuming Hu, Lijie Wen, Irwin King, and Philip S Yu. Can llm watermarks robustly prevent unauthorized knowledge distillation? In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13228–13251, 2025.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

Tom Sander, Pierre Fernandez, Alain Durmus, Matthijs Douze, and Teddy Furon. Watermarking makes language models radioactive. *Advances in Neural Information Processing Systems*, 37: 21079–21113, 2024.

Yash Savani, Asher Trockman, Zhili Feng, Yixuan Xu, Avi Schwarzschild, Alexander Robey, Marc Finzi, and Zico Kolter. Antidistillation sampling. *Advances in Neural Information Processing Systems*, 38:117800–117827, 2026.

Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction {APIs}. In *25th USENIX security symposium (USENIX Security 16)*, pp. 601–618, 2016.

Yihan Wu, Zhengmian Hu, Junfeng Guo, Hongyang Zhang, and Heng Huang. A resilient and accessible distribution-preserving watermark for large language models. *arXiv preprint arXiv:2310.07710*, 2023.

Jiashu Xu, Fei Wang, Mingyu Ma, Pang Wei Koh, Chaowei Xiao, and Muhao Chen. Instructional fingerprinting of large language models. 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. 3277–3306, 2024.

Yixuan Even Xu, John Kirchenbauer, Yash Savani, Asher Trockman, Alexander Robey, Tom Goldstein, Fei Fang, and J Zico Kolter. Antidistillation fingerprinting. *arXiv preprint arXiv:2602.03812*, 2026.

Borui Yang, Mingxuan Ma, Liyao Xiang, Nan Chen, Xin Zhang, Linghe Kong, and Xinghao Jiang. Codegenguard: A watermark for code generation models. In *International Conference on Learning Representations*, volume 2026, pp. 128214–128244, 2026a.

Ziqing Yang, Yixin Wu, Yun Shen, Wei Dai, Michael Backes, and Yang Zhang. The challenge of identifying the origin of black-box large language models. In *Proceedings of the Seventh Workshop on Privacy in Natural Language Processing*, pp. 7–25, 2026b.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 3911–3921, 2018.

Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. Provable robust watermarking for ai-generated text. *arXiv preprint arXiv:2306.17439*, 2023a.

Xuandong Zhao, Yu-Xiang Wang, and Lei Li. Protecting language generation models via invisible watermarking. In *International conference on machine learning*, pp. 42187–42199. PMLR, 2023b.

Zhengyue Zhao, Xiaogeng Liu, Somesh Jha, Patrick McDaniel, Bo Li, and Chaowei Xiao. Can watermarks be used to detect llm ip infringement for free? In *The Thirteenth International Conference on Learning Representations*, 2025.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

A IMPLEMENTATION DETAILS

A.1 EXAMPLES OF SELF-DISTILLED ANSWER PAIRS

Table 6 shows paired training targets produced by the self-distillation procedure in Section 3.2. In some pairs the second answer takes a different route; in others it only rephrases the first.

Table 6: Examples of self-distilled answer pairs used to train the two LoRA adapters. Each row shows the targets $y^{(0)}$ and $y^{(1)}$ generated for the same training prompt. In positive examples the two answers differ in computational route, derivation, or creative focus; in negative examples they share the same core computation or extracted facts. The examples are illustrative and do not estimate prevalence. Family labels record generation order, not answer quality.

Domain and prompt	$y^{(0)}$ **target**	$y^{(1)}$ **target**	**Difference**
Positive: different route or content focus			
Code: minimum coins	Bottom-up dynamic programming over amounts.	Breadth-first search over reachable amounts.	Different search procedures for the same minimum-coin objective.
Math: $4.3 + 3.88$	Align decimal places and add by columns.	Convert to fractions with denominator 100, then add.	Different arithmetic routes; both give 8.18.
Physics: double-slit fringes	Directly substitute into $\Delta y = \lambda L/d$.	Derive fringe spacing from angular maxima and screen geometry.	Derivation versus direct formula; both give 1.2 mm.
Writing: music-production tweet	Focus on a snare hit and balanced levels.	Focus on low-end mixing and patience.	Different creative details; both include hashtags.
Negative: same core method or answer			
Code: even-square/odd-root	Accumulate transformed values in a loop.	Build a list by comprehension, then sum it.	Same transformation and rounding.
Math: 5:6:7 triangle	Set $18x = 180$ and report $50°$.	Rename the multiplier k; set $18k = 180$ and report $50°$.	Same equation and answer.
Chemistry: ice-to-water enthalpy	Apply Kirchhoff's law with the heat-capacity difference.	Apply the same law and numerical correction.	Same method and 6.96 kJ/mol result.
Extraction: Sumners	Name knot theory, topological fluid dynamics, and DNA applications.	Return the same three facts with reordered emphasis.	Same extracted information.

A.2 SEMANTIC BUCKET PREDICTION AND PRIVATE HASHING

The question classifier uses the concatenation of word 1-2 gram and character 3-5 gram TF-IDF features, capped at 4,096 features each. We fit one logistic regression classifier per domain with $C = 4$, balanced class weights, the `lbfgs` solver, and seed 183010. Training targets come from fixed keyword rules. The first matching rule assigns the weak label; these labels are not guaranteed to be accurate subject or intent annotations. Code labels also include the task-source family. At deployment, the fitted classifier reads only the domain and question text: it does not read task IDs, answers, ground truth, reference solutions, generated responses, or detector scores. A classifier cannot predict a bucket absent from its training classes.

Each bucket name combines the domain, the task-source family for code questions, and the predicted category. Weak-label rules follow the category order in Table 7, taking the first match and falling back to `general`. Given the predicted bucket, the bit is computed as

```
bit = hashlib.sha256(
    (secret_key + "\n" + bucket).encode("utf-8")
```

Table 7: Semantic bucket categories. The question classifier predicts these weak-label categories from question text.

Domain	Categories
Code	Source family combined with sequence, string, mapping, structure, numeric, predicate, or general intent.
Math	Age, money, geometry, rate/ratio, count/logic.
Science	Biology, chemistry, earth/space, physics, general.
Text	Translation, summarization, rewriting, extraction, classification, brainstorming, creative writing, practical writing, explanation, general.

```
).digest()[0] & 1
```

The key is fixed without salt search, key resampling, or adjustment based on test scores. Routing and detection use the same question representation. The private key is not included in the question classifier or student training data.

A.3 Detector features

For each response token and reference state, the detector records the observed token's log probability and rank, the top-1 token, and that token's log probability. These observations are summarized in $\phi(x, y, b) \in \mathbb{R}^{98}$: 24 within-state choice statistics, 12 comparisons with the base state, 12 aggregate choice and likelihood statistics, 48 route-sign interactions, response length, and route sign.

A.4 Student training

The selected teacher answers form $\mathcal{D}_k = \{(x_i, y_i)\}_{i=1}^{N}$. Students are trained on the question-answer pairs using

$$\mathcal{L}_{\text{distill}}(S) = -\frac{1}{N} \sum_{i=1}^{N} \log p_S(y_i \mid x_i), \tag{8}$$

where the sequence log-probability decomposes over answer tokens. The student receives no private key, bucket labels, or adapter identities. In the offline selection interface, both teacher answers must already exist; the bit chooses one answer without adding a watermark prefix or rewriting it. Watermark inheritance is evaluated from the trained student's outputs.

Training configuration. All students use LoRA rank $r = 32$, $\alpha = 64$, dropout 0.05, an effective batch size of 16, and a learning rate of 2×10^{-5}. Multi-domain students (Table 3) are trained for one epoch, while MBPP students (Table 4) are trained for 126–314 optimizer updates.

B FULL EXPERIMENTAL RESULTS

Tables 8 and 9 retain every reported teacher and online configuration, including AUROC, detection rate, task utility, and utility difference. The main tables present the teacher utility-detection comparison and common low online budgets; no configurations are discarded.

Table 8: Full teacher results across four benchmarks. Weighted metrics use dataset sizes. TPR is reported at FPR $\leq 1\%$; Δ is the utility difference from Base in percentage points.

Dataset	Method	AUROC (%)	TPR (%)	Utility (%)	Δ (pp)
Math500	Base	—	—	96.00	—
	KGW $\delta = 0.5$	65.30	5.77	96.00	0.00
	KGW $\delta = 1$	76.26	17.52	97.00	+1.00
	KGW $\delta = 2$	88.54	44.02	95.20	−0.80
	KGW $\delta = 3$	95.56	70.51	94.40	−1.60
	KGW $\delta = 5$	99.35	94.02	56.40	−39.60
	DistillTrace	96.32	41.88	96.60	+0.60
APPS	Base	—	—	46.34	—
	KGW $\delta = 0.5$	74.95	15.58	45.52	−0.82
	KGW $\delta = 1$	83.94	39.13	44.54	−1.80
	KGW $\delta = 2$	90.65	59.52	41.58	−4.76
	KGW $\delta = 3$	94.10	68.90	35.44	−10.90
	KGW $\delta = 5$	98.14	88.06	11.26	−35.08
	DistillTrace	98.53	84.46	46.90	+0.56
IFEval	Base	—	—	83.73	—
	KGW $\delta = 0.5$	74.53	13.56	85.40	+1.66
	KGW $\delta = 1$	87.65	39.88	85.03	+1.29
	KGW $\delta = 2$	94.49	75.25	86.14	+2.40
	KGW $\delta = 3$	96.53	87.03	85.03	+1.29
	KGW $\delta = 5$	97.81	93.12	73.57	−10.17
	DistillTrace	97.51	92.93	85.40	+1.66
GPQA Diamond	Base	—	—	51.52	—
	KGW $\delta = 0.5$	79.85	32.53	51.52	0.00
	KGW $\delta = 1$	91.16	61.45	55.56	+4.04
	KGW $\delta = 2$	97.24	87.95	48.99	−2.53
	KGW $\delta = 3$	99.77	95.78	41.41	−10.10
	KGW $\delta = 5$	100.00	100.00	8.59	−42.93
	DistillTrace	100.00	100.00	55.56	+4.04
Weighted	Base	—	—	53.73	—
	KGW $\delta = 0.5$	74.29	15.16	53.21	−0.51
	KGW $\delta = 1$	83.88	38.17	52.60	−1.12
	KGW $\delta = 2$	91.02	60.54	49.98	−3.75
	KGW $\delta = 3$	94.61	71.46	44.65	−9.07
	KGW $\delta = 5$	98.27	89.36	20.20	−33.53
	DistillTrace	98.31	82.28	54.50	+0.77

Table 9: Full online candidate-budget sweep with the same private detector. Aggregate detection averages 256 response scores. TPR is reported at FPR $\leq 1\%$; Δ is the utility difference from the $K = 1$ baseline in percentage points.

				Single response		Aggregate (256)	
Dataset	K	**Utility (%)**	**Δ (pp)**	**AUROC (%)**	**TPR (%)**	**AUROC (%)**	**TPR (%)**
Qwen3.5-9B							
HumanEval	1	88.41	—	—	—	—	—
	2	87.20	−1.22	77.75	10.98	100.00	100.00
	4	84.76	−3.66	92.45	27.44	100.00	100.00
	8	85.98	−2.44	96.10	41.46	100.00	100.00
	16	89.02	+0.61	97.41	58.54	100.00	100.00
MBPP	1	69.80	—	—	—	—	—
	2	71.00	+1.20	71.49	10.00	100.00	100.00
	4	72.40	+2.60	83.55	18.60	100.00	100.00
	8	70.60	+0.80	91.84	28.80	100.00	100.00
	16	69.20	−0.60	96.27	46.80	100.00	100.00
Spider dev	1	71.47	—	—	—	—	—
	2	71.18	−0.29	63.97	3.97	100.00	100.00
	4	69.92	−1.55	76.42	8.41	100.00	100.00
	8	69.15	−2.32	84.44	13.54	100.00	100.00
	16	68.18	−3.29	88.22	19.44	100.00	100.00
Llama 3.1 8B Instruct							
HumanEval	1	68.90	—	—	—	—	—
	2	68.90	0.00	62.29	0.61	100.00	100.00
	4	65.85	−3.05	71.29	3.05	100.00	100.00
	8	63.41	−5.49	80.36	8.54	100.00	100.00
	16	65.24	−3.66	85.82	14.63	100.00	100.00
MBPP	1	58.80	—	—	—	—	—
	2	56.20	−2.60	56.76	1.80	99.81	96.00
	4	55.20	−3.60	64.36	2.40	100.00	100.00
	8	54.00	−4.80	70.14	4.40	100.00	100.00
	16	54.40	−4.40	75.39	6.80	100.00	100.00
Spider dev	1	69.34	—	—	—	—	—
	2	69.73	+0.39	56.30	1.26	99.04	84.44
	4	66.83	−2.51	63.05	2.22	100.00	100.00
	8	66.15	−3.19	69.08	3.00	100.00	100.00
	16	65.67	−3.68	74.61	4.35	100.00	100.00

C Comparison of protection methods

Table 10: Comparison with distillation-tracing methods. DistillTrace is the only method that satisfies all five properties.

		Embedding		Coverage		
Method (venue)	**Signal**	**No proxy student**	**No token bias**	**Answer-only**	**Domain-agnostic**	**Any generator**
KGW-based tracing						
Radioactive (NeurIPS) (Sander et al., 2024)	KGW token bias	●	○	●	●	○
LIDet (ICLR) (Zhao et al., 2025)	KGW token bias	●	○	●	●	○
ADFP (ICML) (Xu et al., 2026)	Proxy-guided KGW	○	○	●	●	○
CoT-based tracing						
Trace rewriting (ACL) (Ma et al., 2026)	Rewritten CoT	○	●	○	◐	●
ReasMark (ACL) (Lv et al., 2026)	CoT length	●	●	○	○	○
Transformation-based tracing						
CodeGenGuard (ICLR) (Yang et al., 2026a)	Code transformation	○	●	●	○	○
DistillTrace (ours)	**Solution preference**	●	●	●	●	●

● supported by design; ◐ restricted setting; ○ not supported. Column definitions follow the table.

Table 10 uses the following column definitions. *No proxy student*: embedding does not optimize against a proxy or surrogate student; ADFP guides sampling with a proxy student, trace rewriting selects rewriting instructions with proxy students, and CodeGenGuard co-trains a shadow model to simulate extraction. *No token bias*: serving requires no decoding-time reweighting of token probabilities. *Answer-only*: the signal is carried by the final answer, so it remains available for distillation and verification when reasoning traces are hidden or discarded; trace rewriting and ReasMark require exposed reasoning traces. *Domain-agnostic*: the signal is not tied to a pre-selected domain; ReasMark selects watermark tokens from high-frequency prompts of a target domain, trace rewriting applies only to tasks with reasoning traces, and CodeGenGuard relies on code-specific semantics-preserving transformations. *Any generator*: the watermark can be added to answers from a generator that is not fine-tuned and whose logits are not modified, including API-only models; CodeGenGuard and ReasMark instead fine-tune the protected model; DistillTrace does so by online selection with GPT-5.6 Luna and Claude Haiku 4.5 (Table 2).