Rethinking AI-Generated Text Detection: A Strong Baseline and the Distribution-Shift Problem That Remains

Anonymous ACL submission

Abstract

Recent AI-generated text detection work often introduces a new benchmark together with a specialized detector tailored to it. We revisit this practice from a baseline-first perspective. Across several benchmarks, we show that a plain, fully fine-tuned RoBERTa matches or exceeds the specialized detectors those benchmarks are built around. This suggests that much of the recent architectural complexity is not what drives strong in-distribution detection. The remaining challenge is the distribution shift. The same strong baseline degrades sharply when the topic domain or generating model changes at test time, and simply adding more source data does not close the gap. We identify a key failure mode: under distribution shift, the detector can assign high-confidence machine labels to human-written text from unseen domains. We then study two lightweight domain adaptation methods to address this problem: K-shot adaptation with first-order MAML over LoRA adapters, and a per-sample confidence-weighted ensemble built on top of the adapted detector. Overall, our results suggest that progress in AI-generated text detection should be measured not only by in-distribution performance, but also by robustness under distribution shift.

1 Introduction

Large language models (LLMs) now generate text that is difficult to distinguish from human writing, raising concerns in settings such as education (Susnjak, 2022), peer review (Yu et al., 2025), and the integrity of online information (Spitale et al., 2023). This has motivated a large and fast-growing body of work on AI-generated text detection.

Recent AI-generated text detection work has continued to introduce new detectors alongside new benchmarks. In several cases, the benchmark is released together with a *specialized* detector reported to outperform prior methods on that benchmark: the IntelLabs peer-review benchmark releases Anchor (Yu et al., 2025), calibrated for strict low-FPR operation; FAID couples a three-way detector with a vector-database augmentation mechanism (Ta et al., 2026); and MIRAGE proposes DetectAnyLLM (Fu et al., 2025). A related line of work builds large "in-the-wild" benchmarks and reports which existing detector performs best on them; MAGE (Li et al., 2024), for instance, finds a long-context pre-trained language model detector to be the strongest among the methods it evaluates.

A striking commonality across these works is the baseline they improve upon: the GPT-2 classifier released by Solaiman et al. (2019), used in a *zero-shot*, un-fine-tuned manner. Such a baseline is useful as an early off-the-shelf reference, but it does not answer the more basic question: how strong is a standard supervised baseline trained on the same data? A zero-shot detector and a classifier fine-tuned on the benchmark's own training set are fundamentally different points of comparison. Without evaluating the latter under the same protocol, it remains unclear whether the reported gains come from detector design or from access to in-distribution training data.

We revisit the problem from this baseline-first perspective in two parts. **First, a strong generic baseline already goes surprisingly far.** We fine-tune RoBERTa (Liu et al., 2019) end-to-end on four detection benchmarks and find that it matches or exceeds the detector each benchmark is built around. As shown in Figure 1 (a), on IntelLabs, a plain fully fine-tuned RoBERTa substantially exceeds the specialized Anchor detector in TPR at 0.1% FPR across five source LLMs. On MAGE, where the strongest reported detector uses a long-context window, a RoBERTa baseline restricted to 512 tokens matches it, indicating that long-context modeling is not necessary to reach the reported detection accuracy.

Second, distribution shift remains the central

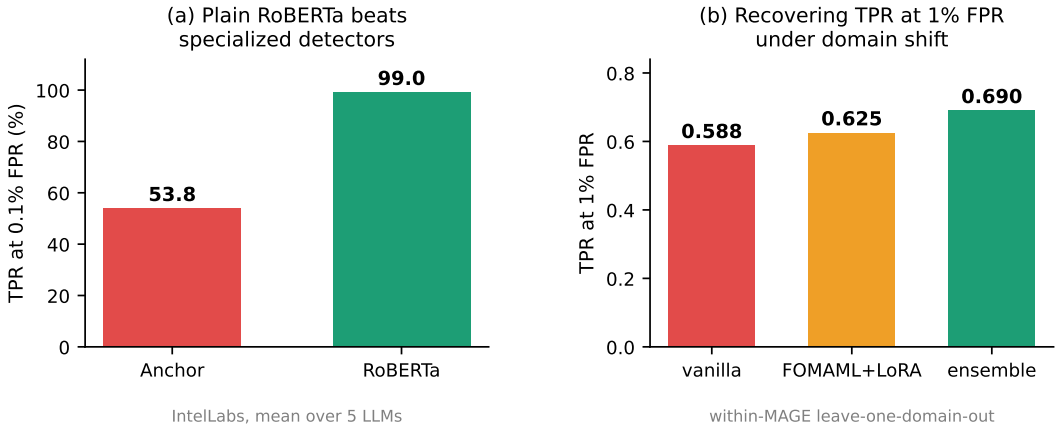

Figure 1: Overview. (a) On IntelLabs peer-review benchmark (Yu et al., 2025), a fully fine-tuned RoBERTa baseline substantially outperforms the specialized Anchor detector over five source LLMs. (b) Under within-MAGE (Li et al., 2024) leave-one-domain-out evaluation, two simple approaches to address distribution shift (FOMAML+LoRA and a confidence-weighted ensemble) both partially recover the TPR at 1% FPR that the vanilla detector loses. The two panels use different benchmarks and FPR thresholds and are not directly comparable.

challenge. A fully fine-tuned detector can be very strong when the test distribution is well covered by the training distribution. In deployment, however, no training set can enumerate every topic domain and every generator a detector may encounter, making robustness to *distribution shift* the more consequential question. We find that a detector with near-perfect in-distribution scores degrades sharply when the topic domain or the generating model changes, and identify a specific failure mode behind this degradation: the detector confidently mislabels a majority of human-written text as machine, a failure that AUROC can obscure but low-FPR evaluation exposes. We then show that two simple approaches partially address this failure mode (Figure 1(b)): a K-shot domain adaptation method using first-order Model-Agnostic Meta-Learning over Low-Rank adapters (FOMAML+LoRA) (Nichol et al., 2018; Hu et al., 2022), and a per-sample confidence-weighted ensemble of the adapted and vanilla detectors.

We summarize our contributions as follows: (1) we show that a fully fine-tuned RoBERTa baseline matches or exceeds the specialized detectors proposed alongside three benchmarks (IntelLabs, FAID, and MIRAGE), and also matches the strongest existing detector reported on the in-the-wild MAGE benchmark once input length is held fixed. This suggests that progress in AI-generated text detection should be evaluated against strong fine-tuned baselines, rather than zero-shot detectors. (2) we identify a key failure mode under distribution shift: strong fine-tuned detectors can assign high-confidence machine labels to human-written text from unseen domains. Low-FPR evaluation exposes this failure, and K-shot adaptation with a confidence-weighted ensemble can partially correct it.

2 Related Work

AI-generated text detection. AI-generated text detection spans zero-shot statistical methods and supervised neural detectors. Zero-shot approaches detect machine-generated text by exploiting likelihood artifacts, curvature, conditional probability, or disagreement patterns (Gehrmann et al., 2019; Solaiman et al., 2019; Mitchell et al., 2023; Bao et al., 2024; Hans et al., 2024). Supervised and hybrid systems introduce richer features, adversarial training, fine-grained labels, or multi-domain benchmarks (Verma et al., 2024; Hu et al., 2023; Li et al., 2024; Abassy et al., 2024; Yu et al., 2025; Ta et al., 2026; Fu et al., 2025). However, this progress has often been measured against fixed or weakly adapted baselines from zero-shot or off-the-shelf settings. Although fine-tuned RoBERTa-style classifiers have appeared in prior studies, they are rarely treated as the main point of comparison under matched data, evaluation protocols, and deployment-relevant operating conditions. In contrast, our work takes a baseline-first perspective: we re-evaluate a fully fine-tuned supervised baseline and measure how much progress remains when the baseline has comparable training access and is evaluated at strict false-positive-rate thresholds.

Table 1: The four AI-generated text detection benchmarks we considered. "Generators" lists either individual LLMs or LLM families (each family covering multiple specific models). Train and Test sizes are in number of examples; Test reports the main in-distribution test split, with additional evaluation splits described in Appendix A.

Dataset	Train	Test	Label	Domains	Generators
IntelLabs (Yu et al., 2025)	23K	192K	binary	1 (peer reviews)	5 LLMs
MAGE (Li et al., 2024)	319K	61K	binary	10 (mixed)	27 LLMs
FAID (Ta et al., 2026)	61K	13K	3-way	2 (academic)	4 families
MIRAGE (Fu et al., 2025)	1K†	188K	binary	5 (mixed)	17 LLMs

†MIRAGE's training pool consists of 500 paired (human-machine) examples, i.e., 1,000 samples total.

Distribution shift, adaptation, and deployment-oriented evaluation. A second line of work concerns the robustness of detectors under distribution shift. Distribution shift is a well-studied problem in machine learning (Koh et al., 2021; Wang et al., 2022), and is especially acute for AI-generated text detection, where the target distribution changes with writing domain, prompt style, paraphrasing, and the underlying LLM. Existing detection benchmarks and recent domain-generalization studies explicitly show that detectors can perform well in-domain but fail to generalize to unseen generators or domains (Li et al., 2024; Borile and Abrate, 2025; Bhattacharjee et al., 2024). We address this setting through domain adaptation, using FOMAML over LoRA adapters (Finn et al., 2017; Nichol et al., 2018; Hu et al., 2022; Houlsby et al., 2019; Li and Liang, 2021), which combines gradient-based meta-learning with parameter-efficient fine-tuning. Beyond adaptation, we use a per-sample confidence-weighted combination of the adapted and vanilla detectors and evaluate it under low-FPR operating points, where the failure mode of interest becomes visible.

3 A Strong Fully Fine-Tuned In-Distribution Baseline

3.1 Problem Setup

We study AI-generated text detection as a binary classification problem: given a piece of input text x, predict whether it is human-written ($y = 0$) or machine-generated ($y = 1$). A detector is a function $f : \mathcal{X} \to [0, 1]$ producing a probability $f(x)$ that x is machine-generated. FAID (Ta et al., 2026) instead uses a three-way label (human/machine/mixed); we train a three-way variant for that comparison as detailed in Appendix A.

This section considers the *in-distribution* setting, where the detector is trained and evaluated on data from the same benchmark. Specifically, each detector is trained on the benchmark's training split and evaluated on its held-out test split, following the benchmark's standard protocol. Our goal is to assess how a standard fully fine-tuned classifier performs under the same conditions in which specialized detectors are typically evaluated.

We evaluate the baseline on four publicly available AI-generated text detection benchmarks (Table 1), covering peer reviews, academic writing, and diverse open-domain genres, with machine-generated text produced by a broad set of LLM families; the same four benchmarks are used throughout the paper, and full details are given in Appendix A. As evaluation metrics, we report AUROC, a threshold-independent summary, together with the true-positive rate at a low, fixed false-positive rate (TPR at 1% FPR, and 0.1% FPR), when a stricter operating point is informative. The low-FPR metric is more relevant for deployment, where detectors use fixed thresholds in practice and falsely flagging human-written text is costly.

3.2 Method: Full Fine-Tuning with RoBERTa

Our baseline detector is deliberately simple: we use RoBERTa (Liu et al., 2019) as the backbone and add a single linear classification head over the `[CLS]` token. The entire backbone is fine-tuned end-to-end, with no frozen layers, auxiliary objectives, or detection-specific architectural components. We train using standard cross-entropy loss against ground-truth labels and optimize with AdamW (Loshchilov and Hutter, 2017). We refer to this detector as the *vanilla* detector throughout the paper.

We use two backbone sizes, RoBERTa-base (125M parameters) and RoBERTa-large (355M parameters), and select the size per benchmark to match the model scale of its reported baseline. The corresponding vanilla detectors are denoted *Vanilla(base)* and *Vanilla(large)* when the backbone size needs to be specified. This isolates the effect of full fine-tuning rather than model capac-

Table 2: IntelLabs peer-review detection on the pre-ChatGPT withheld test set: TPR (%) at different FPR thresholds per source LLM. We compare Anchor, the specialized detector from Yu et al. (2025), against Vanilla, our fully fine-tuned RoBERTa-base baseline trained on the same calibration subset. Anchor numbers from Yu et al. (2025).

	GPT-4o			Gemini-1.5			Claude-3.5			Llama-3.1			Qwen-2.5		
Method	0.1%	0.5%	1.0%	0.1%	0.5%	1.0%	0.1%	0.5%	1.0%	0.1%	0.5%	1.0%	0.1%	0.5%	1.0%
Anchor	63.5	83.7	88.8	59.7	80.3	86.5	59.6	75.8	81.8	13.6	28.5	36.8	72.5	88.4	92.1
Vanilla	**100.00**	**100.00**	**100.00**	**99.99**	**100.00**	**100.00**	**99.98**	**100.00**	**100.00**	**95.37**	**98.74**	**99.39**	**99.85**	**99.96**	**99.99**

Table 3: MAGE detection. We compare the long-context Longformer baseline reported in Li et al. (2024) against Vanilla(large), our fully fine-tuned RoBERTa-large baseline restricted to 512 tokens. The in-distribution setting uses the standard MAGE split with arbitrary domains and arbitrary generators (Appendix A); the paraphrasing-attack setting paraphrases every test input with `gpt-3.5-turbo`, all treated as machine-generated.

	In-distribution		Paraphrasing attack	
Method	AvgRec	AUROC	AvgRec	AUROC
Longformer (MaxLen= 4096)	**90.5**	**0.990**	66.9	0.750
Vanilla(large) (MaxLen= 512)	89.2	0.982	**68.6**	**0.782**

ity. For benchmarks with a binary human/machine label, the head outputs two logits; for the three-way setting of FAID (Ta et al., 2026), the head outputs three, and we recover a binary score $P(\text{machine}) = 1 - P(\text{human})$ from it when comparing against binary detectors. Remaining training details follow standard practice and are reported in Appendix B.

3.3 Full Fine-Tuning Matches or Exceeds Specialized Detectors

We now evaluate how the vanilla detector from Section 3.2 performs in the in-distribution setting, compared with the detector developed in each benchmark. For each of four benchmarks (IntelLabs, FAID, MIRAGE, and MAGE), we reproduce the original paper's in-distribution training and evaluation protocol, i.e., we train on the same data released by the benchmark authors and evaluate on the same test split using the same metrics. This gives a like-for-like comparison between the vanilla detector and the benchmark's reported detector.

IntelLabs peer review. The IntelLabs benchmark (Yu et al., 2025) introduces Anchor, a detector designed for strict low-FPR operation on academic peer reviews. Our Vanilla(base) baseline is trained on exactly the same calibration subset used to set Anchor's threshold, with two of the five source LLMs (Llama-3.1-70B and Qwen-2.5-72B) unseen by both methods during training (Appendix A). Table 2 compares Vanilla(base) with Anchor at the three strict FPR thresholds Anchor is designed for, across the five source LLMs. The gap is large and consistent: Vanilla(base) outperforms Anchor on every LLM at every threshold, with the largest gap at the strictest threshold (0.1% FPR) on the held-out Llama-3.1-70B.

MAGE. MAGE (Li et al., 2024) is a different kind of benchmark: rather than proposing a new detector, it evaluates several existing detectors on a large in-the-wild corpus and reports a long-context Longformer (Beltagy et al., 2020) as the strongest model. Thus, the main comparison is about input length rather than architecture: the MAGE paper attributes Longformer's advantage to its 4,096-token context window, which is eight times longer than RoBERTa's 512-token window. Table 3 compares the 512-token Vanilla(large) with the 4,096-token Longformer in the two settings. The two models perform similarly in distribution, and Vanilla(large) takes the lead under paraphrasing on both AvgRec and AUROC. This is notable because paraphrasing is exactly the setting where a longer context window is expected to help. Overall, the advantage attributed to long-context modeling on MAGE shrinks substantially when compared against a strong 512-token baseline, and even reverses under paraphrasing. Additional results on a further out-of-distribution setting are reported in Appendix C.

FAID. Table 4 compares Vanilla(base) with FAID's own detector on FAID's three-way classification (Ta et al., 2026) across four splits: in-domain and three unseen-domain settings. Even without FAID's vector-database augmentation, Vanilla(base) is essentially tied with FAID on the

Table 4: FAID three-way classification accuracy. We compare two detectors, LLM-DetectAIve (Abassy et al., 2024) and FAID (Ta et al., 2026), against Vanilla and Vanilla + extra, which additionally fine-tunes on each Unseen split's released validation set as a fair analog of FAID's vector-database augmentation. Vanilla reports mean±std; "—" marks splits with no released validation set. LLM-DetectAIve and FAID numbers from Ta et al. (2026).

Method	FAIDSet (in-dom)	Unseen Domain	Unseen Generator	Unseen Domain+Generator
LLM-DetectAIve	94.3	52.8	75.7	62.9
FAID	**95.6**	62.8	**93.3**	66.6
Vanilla	93.7±0.1	72.7±0.8	92.7±1.5	67.7±1.1
Vanilla + extra	—	**91.5±0.2**	—	**88.2±0.5**

in-domain split and outperforms it on both unseen-domain splits, with the larger margin on Unseen Domain. The released validation sets also allow a small amount of additional fine-tuning per split, which is the fair analogue of FAID's vector-database augmentation; using this additional tuning gives the extended model, Vanilla(base) + extra, which further improves Vanilla(base)'s accuracy.

MIRAGE. Table 8 in Appendix C compares our vanilla detectors with DetectAnyLLM on MIRAGE (Fu et al., 2025). All models are trained on the same 500 paired GPT-3.5 polish examples used by DetectAnyLLM. At the same model scale, Vanilla(base) matches DetectAnyLLM in AUROC and falls slightly behind on the other metrics; scaling to Vanilla(large) surpasses DetectAnyLLM on all four metrics. We note that the MIRAGE paper (Fu et al., 2025) also includes a RoBERTa baseline, but it is not fine-tuned: it corresponds to zero-shot inference with the OpenAI 2019 GPT-2 detector (Solaiman et al., 2019), precisely the un-fine-tuned baseline model whose weakness motivates this study.

4 The Distribution-Shift Problem That Remains

Section 3 gives a strong but in-distribution result: a plain fully fine-tuned RoBERTa matches or exceeds specialized detectors when the test data is drawn from the same benchmark it was trained on. The more consequential challenge is the distribution shift. In deployment, a detector is trained once but must handle text from unknown topics and unknown generators; in-distribution performance says little about this setting. We therefore ask the central remaining question: how well does this strong baseline perform when the test distribution shifts away from the training distribution, and what can be done when it fails?

4.1 Distribution Shift Is a Problem In-Distribution Scores Hide

We write $\mathcal{D}_{\text{source}}$ for the labeled distribution used to train a detector and $\mathcal{D}_{\text{target}}$ for the distribution used at evaluation. Section 3 studied the in-distribution case $\mathcal{D}_{\text{source}} = \mathcal{D}_{\text{target}}$; *distribution shift* is the case $\mathcal{D}_{\text{source}} \neq \mathcal{D}_{\text{target}}$.

We evaluate distribution shift under two protocols with different levels of severity. The first is within-MAGE leave-one-domain-out: we train on 9 of MAGE's 10 domains and test on the held-out domain. In this setting, the label space and generator pool are fixed, so only the topic domain changes. This isolates the effect of topic shift. The second is cross-benchmark transfer: we train on one benchmark and evaluate on another, so both the topic domain and the generator distribution may change. For this setting, we also use HC3 (Guo et al., 2023), an early human-versus-ChatGPT corpus covering five English domains, as an additional held-out target benchmark. HC3 is used only as $\mathcal{D}_{\text{target}}$ and is never used to train any detector.

We use within-MAGE to characterize the failure mode, and evaluate the approaches of Section 4.2 under both protocols.

Both distribution-shift protocols show degradation, though with different levels of severity. Cross-benchmark transfer fails most sharply: Table 5 evaluates each single-source detector across different test benchmarks. The diagonal entries give in-distribution performance, while the off-diagonal entries correspond to held-out test benchmarks. At 1% FPR, the off-diagonal entries are far below the diagonal, with several near zero. Within-MAGE leave-one-domain-out is milder but still substantial, with the vanilla detector falling well short of its in-distribution performance across all 10 held-out rounds (Table 6, vanilla column).

Distribution shift persists despite more source data. Simply pooling benchmarks does not fully solve the problem. Under a leave-one-benchmark-

Table 5: Single-source cross-benchmark transfer. Each row is a vanilla detector trained on one benchmark; each column block is a held-out test benchmark, with AUROC and TPR at 1% FPR reported side-by-side. Diagonal cells (in-distribution) shaded. The baseline `OpenAI 2019` (Solaiman et al., 2019) is included for comparison.

	IntelLabs		MAGE		FAID		MIRAGE	
Vanilla trained on	AUROC	TPR	AUROC	TPR	AUROC	TPR	AUROC	TPR
IntelLabs (base)	1.000	0.999	0.533	0.034	0.785	**0.340**	0.752	0.262
MAGE (large)	0.931	0.529	0.982	0.801	0.718	0.038	0.787	0.242
FAID (base)	0.949	0.832	0.551	0.035	0.997	0.924	**0.828**	**0.375**
MIRAGE (large)	**0.995**	**0.891**	0.605	0.119	**0.920**	0.042	0.976	0.784
OpenAI 2019	0.605	0.113	**0.725**	**0.197**	0.459	0.011	0.511	0.029

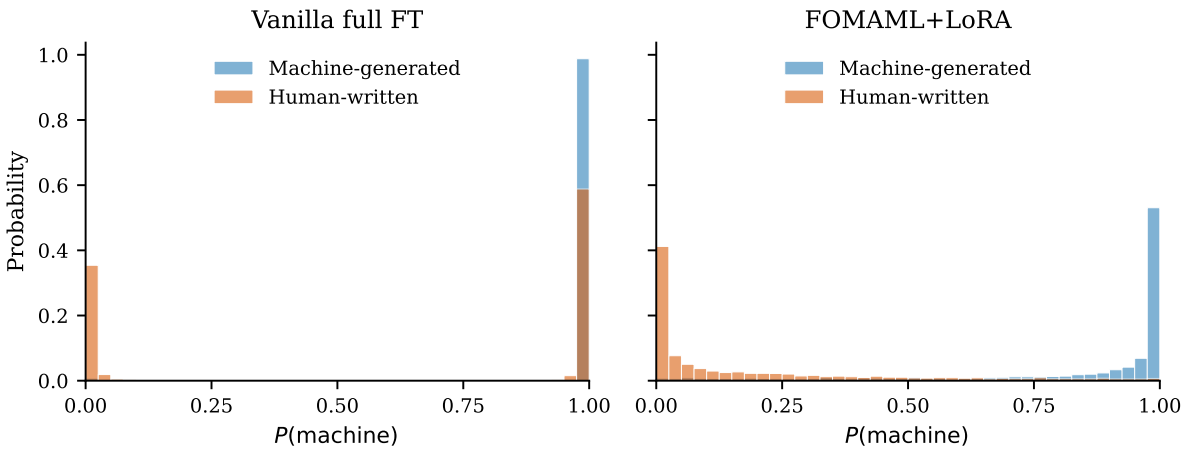

Figure 2: Predicted-probability distributions under within-MAGE leave-one-domain-out, pooled over 10 held-out domains (5,000 machine-generated and 5,000 human-written samples per detector). Left: the vanilla detector is sharply concentrated near 0 and 1, but many human-written samples (orange) are assigned near-certain machine probabilities, overlapping with machine-generated samples (blue). Right: FOMAML+LoRA reduces this failure mode by shifting human-written samples toward 0 and machine-generated samples toward 1.

out protocol, pooling several source benchmarks improves over un-fine-tuned references, but the held-out targets still remain far below their in-distribution performance (Table 11, Appendix D). Since no training set can cover every domain a detector may encounter in deployment, adding more source data alone is not enough to overcome distribution shift.

An overconfidence that AUROC hides. Per-domain AUROC under within-MAGE leave-one-domain-out varies widely across the 10 rounds (Table 6), but the more telling column is the TPR at 1% FPR. The drop from in-distribution performance is much larger at this operating point, so AUROC alone would make the degradation look much less severe. The reason becomes clear from the predicted-probability distributions in Figure 2 (Left). Pooling the 10,000 samples across the 10 held-out domains, 99.4% fall into the highest-confidence bucket, i.e., $p_{\text{vanilla}} \leq 0.05$ or $p_{\text{vanilla}} \geq 0.95$. The detector is almost never uncertain, but its confidence is often misplaced. Within the same high-confidence bucket, the vanilla detector is correct on only 69% of samples: it is not leaving difficult examples in an uncertain middle range, but assigning high-confidence predictions to many errors. The class asymmetry is especially severe. Among the 5,000 human-written samples, 3,019 (60.4%) receive a near-certain machine prediction ($p_{\text{vanilla}} \geq 0.95$). Thus, what appears under AUROC to be a strong detector becomes, at a usable threshold, a detector that confidently labels a majority of human-written text from unseen domains as machine-generated.

4.2 Domain Adaptation Partially Improves Performance

Because adding more source data does not close the gap, we turn instead to domain adaptation with two simple approaches. The first is K-shot domain adaptation (Finn et al., 2017; Nichol et al., 2018; Hu et al., 2022): given a small labeled set from the target domain, we adapt the detector with a few gradient steps before evaluation. The second is a

Table 6: Within-MAGE leave-one-domain-out. Each detector is evaluated on a held-out MAGE domain after training on the other nine. We compare three detectors: Vanilla (our fully fine-tuned RoBERTa-large, no adaptation), FOMAML+LoRA (K=10-shot domain adaptation on the held-out domain), and Ensemble (the per-sample confidence-weighted combination of the two, Eq. 1). AUROC and TPR at 1% FPR are reported per held-out domain. Within each metric group, best per row in **bold**.

	Vanilla		FOMAML+LoRA		Ensemble	
Held-out domain	AUROC	TPR	AUROC	TPR	AUROC	TPR
cmv	0.987	0.866	**0.992**	0.870	0.989	**0.888**
eli5	0.957	**0.662**	0.936	0.484	**0.958**	0.636
hswag	0.930	0.602	0.870	0.584	**0.932**	**0.660**
roct	0.865	0.374	0.867	0.446	**0.907**	**0.532**
sci_gen	0.953	0.720	**0.969**	0.694	0.957	**0.768**
squad	0.975	0.692	0.961	0.720	**0.979**	**0.788**
tldr	0.933	0.626	**0.962**	0.718	0.950	**0.750**
wp	0.971	0.552	**0.982**	0.738	0.980	**0.832**
xsum	0.825	0.274	**0.897**	**0.448**	0.868	0.420
yelp	0.945	0.508	0.944	0.548	**0.950**	**0.624**
Average	0.934	0.588	0.938	0.625	**0.947**	**0.690**

per-sample confidence-weighted ensemble, which combines the vanilla detector with the adapted detector.

Model-Agnostic Meta-Learning (MAML). To make K-shot adaptation efficient for a fully fine-tuned RoBERTa detector, we implement it as first-order MAML over LoRA adapters (FOMAML+LoRA) (Nichol et al., 2018; Hu et al., 2022). FOMAML meta-trains an initialization that can adapt quickly from a small labeled set. Unlike full MAML, it uses the gradient computed after adaptation as the meta-gradient and avoids second-order Hessian-vector products, making meta-training substantially cheaper. LoRA further improves efficiency by updating only a small set of trainable parameters inserted into the backbone, while keeping the pre-trained weights frozen.

In our setting, each source domain is treated as a task. During meta-training, the LoRA adapters are optimized so that a few gradient steps on K labeled examples per class from any source domain produce a good detector for that domain. At test time, we apply the same adaptation procedure using $K = 10$ labeled examples per class from the target domain, starting from the meta-learned initialization. Full equations and hyperparameters are given in Appendix B.

Ensemble Approach. The two detectors have very different prediction profiles (Figure 2): the vanilla detector concentrates almost all probability mass near 0 or 1, while FOMAML+LoRA spreads its predictions across the interval. This contrast motivates using the vanilla detector itself as the confidence signal: distance from 0.5 is a meaningful indicator of (over)confidence for the vanilla detector but not for the adapted one. We therefore combine the two detectors as

$$\hat{p}(x) = w(x)\, p_{\text{vanilla}}(x) + \big(1 - w(x)\big)\, p_{\text{FOMAML}}(x), \tag{1}$$

where the weight function $w(x) \in [0, 1]$ can be chosen to control how much the ensemble trusts the vanilla detector on each input. We considered several simple choices, including fixed linear weights $\alpha\, p_{\text{vanilla}}(x) + (1 - \alpha)\, p_{\text{FOMAML}}(x)$ with $\alpha \in \{0.1, 0.2, \ldots, 0.9\}$, as well as confidence-based variants that use either detector as the confidence source. The best-performing choice among these strategies is

$$w(x) = 2\,|p_{\text{vanilla}}(x) - 0.5| \in [0, 1].$$

Fixed linear combinations never exceed the better single detector by more than a small margin (Table 12), and using FOMAML+LoRA as the confidence source leads to substantially worse performance at 1% FPR (Table 13). We therefore report Eq. 1 with the vanilla-confidence weight as the empirical winner among the ensemble strategies we tried.

Both approaches improve performance under distribution shift. We evaluate the methods under the within-MAGE leave-one-domain-out setting of Section 4.1: detectors are trained on 9 source domains and tested on the held-out domain. Table 6 compares the vanilla detector, FOMAML+LoRA adaptation, and the confidence-weighted ensemble built on top of the adapted detector across the 10

Table 7: MAGE→HC3 cross-benchmark transfer. The same three detectors as Table 6, evaluated under cross-benchmark distribution shift. Vanilla is trained on MAGE and applied zero-shot on each HC3 domain; FOMAML+LoRA performs K=10-shot adaptation per HC3 domain (averaged over 5 random support sets); Ensemble combines them per Eq. 1. AUROC and TPR at 1% FPR are reported per HC3 domain. Within each metric group, best per row in **bold**.

	Vanilla		FOMAML+LoRA		Ensemble	
HC3 domain	AUROC	TPR	AUROC	TPR	AUROC	TPR
finance	**0.997**	**0.972**	0.998	0.922	0.997	0.972
medicine	0.976	0.820	**0.991**	0.813	0.979	**0.880**
open_qa	0.977	0.888	0.964	0.642	**0.978**	**0.894**
reddit_eli5	0.997	0.924	0.998	0.966	0.997	**0.968**
wiki_csai	0.992	0.875	**0.998**	**0.958**	0.995	0.893
Average	0.988	0.896	0.989	0.860	**0.989**	**0.921**

held-out MAGE domains. FOMAML+LoRA improves over the vanilla baseline at 1% FPR, and the ensemble further improves performance by combining the vanilla and adapted detectors. The gains are modest in AUROC but substantial at the low-FPR operating point, where the distribution-shift failure is most visible. The ensemble improves over vanilla on all 10 held-out domains, with the largest gains on the domains where vanilla performs worst.

Improvement extends to cross-benchmark transfer. The within-MAGE setting keeps the generator pool fixed and changes only the topic domain. We now test a harder cross-benchmark transfer setting, where both the topic domain and the generator distribution change. Specifically, we train the vanilla detector on MAGE and evaluate it on the five English HC3 domains (Guo et al., 2023): finance, medicine, open-domain QA, Reddit, and Wikipedia. The vanilla detector is applied zero-shot, while FOMAML+LoRA performs K=10-shot adaptation separately for each HC3 domain. Table 7 reports per-domain AUROC and TPR at 1% FPR for the vanilla detector, FOMAML+LoRA, and the confidence-weighted ensemble. The ensemble again improves over the vanilla baseline at 1% FPR, with the largest gain on the medicine domain. The adaptation is also insensitive to the number of labeled target examples: varying K over $\{5, 10, 20, 50\}$ leaves both AUROC and TPR at 1% FPR essentially unchanged (Table 10, Appendix D). This suggests that the gain mainly comes from the meta-learned initialization, rather than from the particular labeled examples used for adaptation.

5 Conclusion

We revisited AI-generated text detection from a baseline-first perspective, and the first lesson is both simple and consequential: strong general-purpose methods can outperform specialized hand-engineered designs. A plain, fully fine-tuned RoBERTa matches or exceeds the specialized detectors that recent benchmarks are built around, and on the in-the-wild MAGE benchmark, it matches the strongest reported detector once input length is controlled for. This means that a strong and valid baseline is not secondary, but a prerequisite for meaningful innovation. When progress is measured against a zero-shot, un-fine-tuned reference point, that progress has likely been overstated.

At the same time, a strong baseline does not make the problem solved. It degrades sharply once the test distribution moves outside the training distribution, and simply accumulating more source data does not close the gap. What helps is domain adaptation: K-shot meta-learning with FOMAML+LoRA improves performance from only a small labeled target set, and a per-sample confidence-weighted ensemble of the adapted and vanilla detectors provides further gains. This suggests that the remaining bottleneck is not only better detector design, but better evaluation: without testing distribution shift against strong baselines, it is hard to tell whether a method has solved the real deployment problem or only improved an easier in-distribution one.

Taken together, these findings suggest that progress in AI-generated text detection should be measured against strong fine-tuned baselines, not weak off-the-shelf references. Such baselines do not diminish innovation; they make clear where innovation is actually needed: improving robustness when the distribution changes and ensuring reliable behavior at the low false-positive rates required in deployment.

Limitations

Input-length truncation. Our detectors truncate inputs to 512 tokens. On the benchmarks we evaluate this does not appear to cost detection accuracy. Under the controlled 512-token comparison on MAGE, the vanilla detector is competitive with a long-context detector even under paraphrasing attack (Section 3.3). Truncation does, however, leave an adversarial surface our experiments do not cover: an input whose first 512 tokens are human-written, with machine-generated content placed only afterward, would be judged on its human prefix alone and could evade detection. A detector intended to be robust to this would need to process the full input.

Binary formulation of the ensemble. The confidence-weighted ensemble is defined for binary detection: its weight $w(x) = 2\,|p_{\text{vanilla}}(x) - 0.5|$ is built around the single decision boundary at 0.5. It does not directly extend to multi-class formulations such as the three-way human/machine/mixed setting of FAID, where no single such boundary exists. A multi-class version of the confidence weighting is left to future work.

Deployment cost and supervision. Both proposed methods carry deployment assumptions. The ensemble runs two detectors, and so requires a second forward pass at inference. The K-shot adaptation assumes a small set of labeled examples from the target distribution; while $K = 10$ per class suffices in our experiments, settings with no labeled target data fall outside what it addresses.

The ensemble's gain is conditional on the failure mode being present. The confidence-weighted ensemble we report is the empirical winner among the simple strategies we tried in the settings of Section 4.2, where the vanilla detector exhibits a large high-confidence false-positive tail. We have not established that this ranking holds in regimes where the vanilla detector is already well-behaved on the negative class. The ensemble should be read as an empirical observation tied to the failure mode characterized in Section 4.1, not as a generic detector-combination method.

References

Mervat Abassy, Kareem Elozeiri, Alexander Aziz, Minh Ngoc Ta, Raj Vardhan Tomar, Bimarsha Adhikari, Saad El Dine Ahmed, Yuxia Wang, Osama Mohammed Afzal, Zhuohan Xie, and 1 others. 2024. LLM-DetectAIve: A tool for fine-grained machine-generated text detection. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 336–343.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. 2024. Fast-DetectGPT: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Amrita Bhattacharjee, Raha Moraffah, Joshua Garland, and Huan Liu. 2024. EAGLE: A domain generalization framework for AI-generated text detection. *arXiv preprint arXiv:2403.15690*.

Claudio Borile and Carlo Abrate. 2025. How to generalize the detection of AI-generated text: Confounding neurons. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 25461–25476, Suzhou, China. Association for Computational Linguistics.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1126–1135. PMLR.

Jiachen Fu, Chun-Le Guo, and Chongyi Li. 2025. DetectAnyLLM: Towards generalizable and robust detection of machine-generated text across domains and models. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 11229–11238.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116, Florence, Italy. Association for Computational Linguistics.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is ChatGPT to human experts? comparison corpus, evaluation, and detection. *arXiv preprint arXiv:2301.07597*.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2024. Spotting LLMs with binoculars: Zero-shot detection of machine-generated text. In *Proceedings of the 41st International Conference on Machine Learning*, pages 17519–17537. PMLR.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, pages 2790–2799. PMLR.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. 2023. RADAR: Robust AI-text detection via adversarial learning. In *Advances in Neural Information Processing Systems*.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Sara Beery, Jure Leskovec, Anshul Kundaje, Emma Pierson, Sergey Levine, Chelsea Finn, and Percy Liang. 2021. WILDS: A benchmark of in-the-wild distribution shifts. In *Proceedings of the 38th International Conference on Machine Learning*, pages 5637–5664. PMLR.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, pages 4582–4597. Association for Computational Linguistics.

Yafu Li, Qintong Li, Leyang Cui, Wei Bi, Zhilin Wang, Longyue Wang, Linyi Yang, Shuming Shi, and Yue Zhang. 2024. MAGE: Machine-generated text detection in the wild. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 36–53.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, pages 24950–24962. PMLR.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, and 1 others. 2019. Release strategies and the social impacts of language models. *arXiv preprint arXiv:1908.09203*.

Giovanni Spitale, Nikola Biller-Andorno, and Federico Germani. 2023. AI model GPT-3 (dis)informs us better than humans. *Science Advances*, 9(26):eadh1850.

Teo Susnjak. 2022. ChatGPT: The end of online exam integrity? *arXiv preprint arXiv:2212.09292*.

Minh Ngoc Ta, Dong Cao Van, Duc-Anh Hoang, Minh Le-Anh, Truong Nguyen, My Anh Tran Nguyen, Yuxia Wang, Preslav Nakov, and Dinh Viet Sang. 2026. FAID: Fine-grained AI-generated text detection using multi-task auxiliary and multi-level contrastive learning. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3275–3296.

Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. 2024. Ghostbuster: Detecting text ghostwritten by 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)*, pages 1702–1717, Mexico City, Mexico. Association for Computational Linguistics.

Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip S. Yu. 2022. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):8052–8072.

Sungduk Yu, Man Luo, Avinash Madasu, Vasudev Lal, and Phillip Howard. 2025. Is your paper being reviewed by an LLM? benchmarking AI text detection in peer review. *arXiv preprint arXiv:2502.19614*.

A Dataset Details

This appendix expands the dataset overview of Section 3 and Table 1, describing the domains, generators, and train/test protocols of each benchmark.

IntelLabs peer review. The IntelLabs benchmark (Yu et al., 2025) consists of academic peer reviews from two venues, ICLR and NeurIPS. Human-written reviews are treated as reliably human only up to 2022; from 2023 onward they may contain AI-assisted content, so we restrict all experiments to the pre-ChatGPT portion. The dataset provides three paper-level splits that are disjoint by `unique_paper_id` (verified to have zero overlap): a *calibration* split (ICLR and NeurIPS 2021–2022, $2{,}000$ unique papers), a *test* split (the pre-ChatGPT withheld set, $5{,}406$ unique papers across multiple years), and an *extended* split. Machine-generated reviews are produced by five source LLMs: GPT-4o, Gemini-1.5-Pro, Claude-Sonnet-3.5, Llama-3.1-70B, and Qwen-2.5-72B. We train only on the calibration split, dividing it $90/10$ into training and validation via `GroupShuffleSplit` grouped on `unique_paper_id`; grouping by paper, rather than by individual review, prevents multiple reviews of the same paper from being split across training and validation. Anchor itself has no trainable parameters: it learns only a per-LLM threshold on the calibration set, using reference reviews generated from GPT-4o, Gemini-1.5-Pro, and Claude-Sonnet-3.5. Our Vanilla(base) baseline is trained on the same calibration subset (ICLR reviews from these three LLMs), so that the comparison in Section 3.3 holds Llama-3.1-70B and Qwen-2.5-72B out of training for both methods.

MAGE. MAGE (Li et al., 2024) is a 437K-example corpus spanning ten topic domains (`cmv`, `eli5`, `hswag`, `roct`, `sci_gen`, `squad`, `tldr`, `wp`, `xsum`, and `yelp`), with machine text from twenty-seven generators. The original paper defines eight testbeds of increasing distributional difficulty: Testbeds 1–4 are in-distribution (training and test drawn from the same sources, varying whether the domain and the generator set are fixed or arbitrary), and Testbeds 5–8 are out-of-distribution (test data from unseen models, unseen domains, or both). We evaluate on Testbed 4 (arbitrary domains and arbitrary models, the most comprehensive in-distribution setting), Testbed 7 (four held-out datasets, namely CNN/DailyMail, DialogSum, PubMedQA, and IMDb, with GPT-4-generated machine text), and Testbed 8 (Testbed 7 with every text additionally paraphrased by `gpt-3.5-turbo`, all treated as machine-generated). We focus on these three because they are released as ready-to-use splits, they share a single training set so that one trained model evaluates all three, and they cover the most informative in-distribution, out-of-distribution, and adversarial regimes; Testbeds 1–3 show little separation between methods, and Testbeds 5–6 require leave-one-out cross-validation with many separately trained classifiers.

For the within-MAGE leave-one-domain-out experiments (Section 4.2), each domain is split $80/20$ with a fixed seed; the evaluation portion of each class is capped at $\min(500, 20\%)$ examples. In each round, the held-out domain contributes only its evaluation portion as the test set, while the remaining nine domains contribute only their training portions to the pooled training set. The split of each domain is identical across all rounds, so no test example of a held-out domain ever appears in another round's training pool.

FAID. FAID (Ta et al., 2026) is an academic-text benchmark with a three-way label space: human-written, fully machine-generated, and human-machine collaborative. Its training set, FAID-Set/train, covers two academic domains, student theses and paper abstracts, with machine text from four LLM families (GPT, Gemini, DeepSeek, and Llama). FAID defines four evaluation splits according to whether the topic domain and the generator are seen during training: *in-domain* (same domains and generators as training), *Unseen Domain* (IELTS essays, with the four training generators), *Unseen Generator* (academic text, with three unseen generators: Qwen, Mistral, and Gemma), and *Unseen Domain + Generator* (IELTS essays with the three unseen generators). The Unseen Generator split is released with a test set only and no validation split; consequently, the additional-fine-tuning variant we report on the other unseen splits cannot be applied to it. We use the English portion of FAID only.

MIRAGE. MIRAGE (Fu et al., 2025) is sampled from five text domains, with machine text from seventeen LLMs. Its detector, DetectAnyLLM, is trained on 500 paired GPT-3.5 polish examples, which we use as the training data for our vanilla detectors to enable a fair comparison. The test set comprises six subsets, formed by crossing two input settings with three revision tasks. The two

input settings are Shared-Input Generation (SIG), in which human and machine texts share the same input prompt, and Disjoint-Input Generation (DIG), in which they do not. The three revision tasks are Generate, Polish (small-scale stylistic revision of existing human text), and Rewrite. Because the SIG subsets are class-imbalanced, we additionally report balanced accuracy and MCC alongside AUROC for MIRAGE.

HC3. HC3 (Guo et al., 2023) is an early human-versus-ChatGPT comparison corpus of question-answering text, with ChatGPT as the single generator. We use its five English domains: finance, medicine, open_qa, reddit_eli5, and wiki_csai. HC3 is not used to train any source detector; it serves only as a target distribution for the MAGE→HC3 cross-benchmark experiments, where the vanilla detector is evaluated zero-shot and FOMAML+LoRA performs K-shot adaptation using a small support set drawn from each HC3 domain.

B Training Details

This appendix gives the training and optimization details omitted from Sections 3 and 4.

Vanilla detector. All vanilla detectors are fine-tuned end-to-end with AdamW, a learning rate of 2×10^{-5} for Vanilla(base) and 1×10^{-5} for Vanilla(large), a batch size of 32, 3 epochs, a warmup ratio of 0.06, and a maximum input length of 512 tokens; inputs longer than 512 tokens are truncated. The same configuration is used for every benchmark in Section 3 and for the single-source and pooled detectors in Section 4; only the backbone size varies, selected per benchmark as described in Section 3.2.

FOMAML+LoRA. The LoRA adapters use rank $r = 16$, scaling factor $\alpha = 32$, and dropout 0, and are inserted into the query and value projections of every self-attention layer of Vanilla(large); the backbone is frozen. This yields approximately 2.6M trainable parameters, about 0.7% of the full model.

Let θ collect the trainable parameters (the LoRA adapters and the classification head). Each source domain is treated as a task. Algorithm 1 summarizes the full procedure. At every meta-iteration we sample a batch of tasks; for a task $\mathcal{T}_i$ with a support set S_i and a query set Q_i, we run n steps

Algorithm 1 FOMAML+LoRA meta-training and test-time adaptation

Require: Source tasks $\{\mathcal{T}_1, \ldots, \mathcal{T}_T\}$; target task $\mathcal{T}_{\text{target}}$; $\eta_{\text{meta}}, \eta_{\text{inner}}, n, K, M$

Initialize θ (LoRA adapters + head); freeze backbone
for iter $= 1, \ldots, M$ **do** ▷ Meta-training
  Sample task batch from $\{\mathcal{T}_i\}$; for each $\mathcal{T}_i$ sample S_i, Q_i
  For each $\mathcal{T}_i$: $\theta_i^{(n)} \leftarrow \textsc{InnerLoop}(\theta, S_i, n)$, $g_i \leftarrow \nabla_{\theta_i^{(n)}} \mathcal{L}_{Q_i}(\theta_i^{(n)})$
  $\theta \leftarrow \theta - \eta_{\text{meta}} \cdot \text{Adam}(\frac{1}{|\text{batch}|} \sum_i g_i)$
end for
return $\theta_{\text{target}}^{(n)} \leftarrow \textsc{InnerLoop}(\theta, S_{\text{target}}, n)$ ▷ Test-time adaptation on $\mathcal{T}_{\text{target}}$

function $\textsc{InnerLoop}(\theta_0, S, n)$
  $\theta^{(0)} \leftarrow \theta_0$
  for $s = 0, \ldots, n-1$ **do**
    $\theta^{(s+1)} \leftarrow \theta^{(s)} - \eta_{\text{inner}} \nabla_\theta \mathcal{L}_S(\theta^{(s)})$
  end for
  return $\theta^{(n)}$
end function

of inner-loop SGD on the support set,

$$\theta_i^{(s+1)} = \theta_i^{(s)} - \eta_{\text{inner}} \nabla_\theta \mathcal{L}_{S_i}\left(\theta_i^{(s)}\right), \qquad (2)$$

starting from $\theta_i^{(0)} = \theta$ and producing adapted parameters $\theta_i^{(n)}$. First-order MAML then uses the gradient evaluated *at* the adapted parameters as the meta-gradient, applied with Adam (Kingma and Ba, 2014):

$$\theta \leftarrow \theta - \eta_{\text{meta}} \cdot \text{Adam}\left(\frac{1}{T} \sum_{i=1}^{T} \nabla_{\theta_i^{(n)}} \mathcal{L}_{Q_i}\left(\theta_i^{(n)}\right)\right), \qquad (3)$$

which avoids the second-order Hessian-vector products of full MAML at little cost in performance (Nichol et al., 2018).

Meta-training runs for 2000 meta-iterations with an outer learning rate $\eta_{\text{meta}} = 10^{-4}$ (Adam); each task uses $K = 10$ support examples per class, $|Q_i| = 16$ query examples per class, and $n = 3$ inner-loop SGD steps with inner learning rate $\eta_{\text{inner}} = 10^{-3}$. At test time, given an unseen target domain, we draw a support set of K labeled examples per class from its training pool, run the same n inner-loop steps from the meta-learned θ, and evaluate the adapted detector on the target test set.

Table 8: MIRAGE detection, averaged across six task subsets (DIG/SIG $\times$ generate/polish/rewrite). Our vanilla detectors are fine-tuned on the 500 paired GPT-3.5 polish examples and report mean$\pm$std over 3 seeds.

Method	AUROC	BAcc	MCC	TPR@5%
RoBERTa-base (OpenAI 2019)	0.510	0.516	0.044	0.074
RoBERTa-large (OpenAI 2019)	0.515	0.526	0.053	0.080
DetectAnyLLM	0.934	0.881	0.764	0.773
Vanilla(base)	0.934$\pm$0.010	0.871$\pm$0.010	0.746$\pm$0.017	0.767$\pm$0.018
Vanilla(large)	**0.963$\pm$0.009**	**0.909$\pm$0.019**	**0.820$\pm$0.038**	**0.859$\pm$0.042**

Table 9: MAGE additional out-of-distribution evaluation on a held-out split (CNN/DailyMail, DialogSum, PubMedQA, IMDb, with GPT-4-generated machine text), changing both topic domain and generator relative to training.

Method	AvgRec	AUROC
Longformer (paper, 4096)	75.8	0.940
Vanilla(large) (ours, 512)	**76.0**	**0.969**

Table 10: K-shot sensitivity analysis on MAGE$\rightarrow$HC3 cross-benchmark transfer. FOMAML+LoRA performance averaged across the five HC3 domains (finance, medicine, open_qa, reddit_eli5, wiki_csai), with K varying over $\{5, 10, 20, 50\}$. Each entry is averaged over 5 random support sets per domain. Variation across K is small in both metrics, supporting the claim that the meta-learned initialization, not the particular support examples, carries the adaptation.

K	AUROC	TPR @ 1% FPR
5	0.989	0.873
10	0.989	0.864
20	0.990	0.877
50	0.989	0.873

Every reported FOMAML+LoRA number is averaged over 5 independently drawn random support sets to control for support-set variance.

Random seeds. The in-distribution benchmark comparisons of Section 3 that report error bars (Tables 4 and 8) are averaged over 3 random seeds (42, 123, 2024). The cross-benchmark and within-MAGE adaptation experiments of Section 4 use a single fixed seed (42), as their compute cost makes repeated runs impractical; tables reporting these results are marked accordingly.

Data leakage control. For the pooled cross-benchmark training of Section 4.1, we remove any training example sharing a 30-token n-gram with a held-out test example, so that reported transfer numbers are not inflated by overlap between the pooled sources and the target test set.

C Additional Results for Section 3.3

This appendix reports the full per-benchmark results for MIRAGE (Table 8) summarized in Section 3.3, together with an additional MAGE out-of-distribution evaluation (Table 9).

MAGE held-out domains. Beyond the in-distribution and paraphrasing-attack settings of Table 3, MAGE provides a further out-of-distribution split with four held-out datasets (CNN/DailyMail, DialogSum, PubMedQA, IMDb) paired with GPT-4-generated machine text, changing both topic domain and generator at once. Table 9 shows a fully fine-tuned Vanilla(large) operating on 512 tokens matches the long-context Longformer on AvgRec (76.0 vs 75.8) and exceeds it on AUROC (0.969 vs 0.940), consistent with the in-distribution and paraphrasing-attack patterns of Section 3.3.

D Additional Results for Section 4.2

This appendix reports additional results for Section 4.2: the K-shot sensitivity analysis on MAGE$\rightarrow$HC3 transfer (Table 10), the multi-source pooling under leave-one-benchmark-out (Table 11), the fixed mixing coefficient versus per-sample weighting comparison (Table 12), and the comparison between symmetric ensemble designs (Table 13).

Table 11: Multi-source pooling under leave-one-benchmark-out evaluation (AUROC). "4-way Mixed" merges three non-held-out benchmarks via per-benchmark capping; "Stratified base/large" uses six-super-domain stratified sampling. "Best Single" is the strongest cross-benchmark detector for that column from Table 5.

Held-out test	4-way Mixed	Strat base	Strat large	Best Single
IntelLabs	0.968	0.970	**0.997**	0.995
MAGE	0.592	0.565	0.610	**0.725**[†]
FAID	0.764	0.779	0.828	**0.920**
MIRAGE	0.777	0.827	**0.868**	0.828

[†]Best Single for MAGE is OpenAI 2019 detector (Table 5), which benefits from GPT-2-era LLM overlap with MAGE training data.

Table 12: Fixed mixing coefficient versus the per-sample confidence-weighted ensemble, on within-MAGE leave-one-domain-out (averaged over 10 held-out domains). Each row mixes the two detectors as $p = \alpha\, p_{\text{vanilla}} + (1 - \alpha)\, p_{\text{FOMAML}}$ with a constant α. The confidence-weighted ensemble (Eq. 1) uses the per-sample weight $w(x)$ instead.

Mixing	AUROC	TPR@1%FPR
$\alpha = 0.00$ (FOMAML+LoRA)	0.938	0.625
$\alpha = 0.50$	0.941	0.629
$\alpha = 0.70$	0.941	0.631
$\alpha = 0.80$	0.942	0.632
$\alpha = 0.85$	0.942	0.635
$\alpha = 0.90$	0.942	0.636
$\alpha = 1.00$ (vanilla)	0.934	0.588
Confidence-weighted	**0.947**	**0.690**

Table 13: Two symmetric ensemble designs, evaluated on within-MAGE leave-one-domain-out (10-domain average). The two designs differ only in which detector acts as the dominant predictor and as the confidence source for the per-sample weight w.

Ensemble	AUROC	TPR @ 1% FPR
$w \cdot p_{\text{vanilla}} + (1 - w) \cdot p_{\text{FOMAML}}, \quad w = 2\,\lvert p_{\text{vanilla}} - 0.5 \rvert$	**0.947**	**0.690**
$w \cdot p_{\text{FOMAML}} + (1 - w) \cdot p_{\text{vanilla}}, \quad w = 2\,\lvert p_{\text{FOMAML}} - 0.5 \rvert$	0.928	0.497