MSDOS: Multi-Style Distillation Observers for Robust AI-Generated Text Detection

Anonymous ACL submission

Abstract

Recent advances in large language models have amplified concerns about detecting AI-generated text in real-world settings. Most existing zero-shot detectors, however, implicitly assume a single known generator and a fixed decoding distribution, which often fails under unseen models, mixed sources, and decoding-induced shifts. We present **MSDOS (Multi-Style Distillation Observers)**, a robust zero-shot detector that constructs an AI-side reference distribution from multiple distilled style observers, enabling reliable detection under unknown and mixed generators. MSDOS distills diverse generation behaviors into a set of style observers implemented as lightweight adapters on a shared backbone language model, and aggregates them via likelihood-based fusion by adaptively weighting observers according to their probabilistic consistency with the input. To further handle decoding variations, we introduce a repetition-penalty compensation mechanism that mitigates distribution shifts caused by repetition-penalized generation. Extensive experiments show that MSDOS consistently outperforms prior zero-shot detectors across unseen generators, domains, and decoding settings. Code and data are available at https://github.com/anonymacl/MSDOS.

1 Introduction

With the rapid advancement of large language models (LLMs) (OpenAI et al., 2024; Touvron et al., 2023; Bai et al., 2023), AI-generated text has become pervasive in applications such as news writing, academic assistance, and social media (Sun et al., 2025). While LLMs greatly improve the efficiency of content creation, their low barrier to use also introduces risks including misinformation dissemination (Gehrmann et al., 2019) and academic misconduct (Holmes et al., 2023), making reliable detection of AI-generated text in real-world settings increasingly important (Yang et al., 2024b).

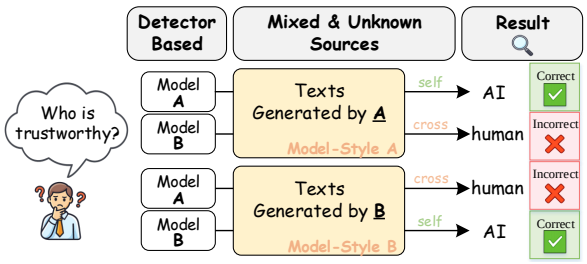

Figure 1: A single observer calibrated to one model-style can fail on other styles: aligned styles produce correct decisions, but cross-style mismatch induces errors, raising trustworthiness concerns under unknown/mixed sources.

Recent zero-shot and unsupervised detection methods typically rely on probability-based analysis using external language models, such as likelihood or perturbation-consistency criteria (Ghosal et al., 2023). While effective under the assumption of a single known generator with fixed decoding strategies, these methods often degrade in realistic settings, where AI-generated text originates from unknown and mixed generators and is produced under diverse decoding configurations (Wu et al., 2024a). In particular, many probability-based detectors rely on a single external language model as an AI-side reference to score text. When the test-time generator differs in style or decoding configuration, such single-reference designs can become miscalibrated and lead to unstable detection under unknown or mixed sources (Figure 1).

Beyond generator heterogeneity, decoding strategies introduce another major source of distribution shift. Common decoding configurations, including temperature, top-k/top-p sampling, and repetition penalty, can systematically reshape token-level probability structures and repetition patterns (Dugan et al., 2024). As a result, detectors that rely on a fixed AI-side reference or static statistical cues often become brittle: cross-generator style mismatch can cause systematic errors, and

even when the generator is unchanged, altering decoding hyperparameters can noticeably shift detection scores and lead to inconsistent decisions. ***How can we make zero-shot AI-text detection reliable under open-world distribution shifts?***

To address this challenge, we propose MSDOS (Multi-Style Distillation Observers), a unified zero-shot detection framework for realistic deployment under open-world generator and decoding shifts. We view a model-style as a systematic preference over possible continuations, which manifests as characteristic token-level probability patterns and likelihood profiles. Instead of assuming a single fixed generator or a single reference view of "AI-like" text, MSDOS distills multiple complementary styles into lightweight observers on a shared backbone language model. These observers are then aggregated into a unified AI-side reference distribution via sentence-level likelihood-based weighting. This design improves robustness under unknown and mixed generators by emphasizing observers that best explain the input text.

Our contributions. We present **MSDOS**, a multi-style distillation framework that scales to heterogeneous generators using only unlabeled AI-generated text. We propose an adaptive fusion strategy that aggregates multiple style observers into a more representative AI-side reference distribution, improving stability under unknown and mixed sources. We further introduce a repetition-penalty compensation mechanism to counter decoding-induced distribution shifts, enhancing robustness across diverse generation settings.

2 Related Work

AI-generated text detection It is typically formulated as binary classification, distinguishing human-written from model-generated text (Richburg et al., 2024). While some methods assume access to the generator (or a closely related LM), real deployments often involve black-box and evolving generators, providing only output texts (Zhang et al., 2024). This has shifted the focus from single-known-generator assumptions toward robustness under unknown and mixed sources (Wu et al., 2024a; Wang et al., 2024), where generator heterogeneity and decoding-/attack-induced distribution shifts are central challenges.

Supervised Detection Methods. Supervised detectors train discriminative classifiers on labeled human–AI text pairs generated by specific models (Liu et al., 2023; Chen et al., 2023; Sarvazyan et al., 2023). Although they perform well when train–test distributions match (Wang et al., 2023; Li et al., 2024), their accuracy often drops under generator, domain, or language shifts (Kuznetsov et al., 2024). Moreover, decoding variations and lightweight post-processing can substantially change surface statistics, further hurting robustness. These limitations motivate zero-shot and unsupervised paradigms that avoid labeled data and do not assume knowledge of target generators.

Zero-Shot and Perturbation-Based Methods. Zero-shot detection methods generally rely on external language models and exploit probability-derived statistics to distinguish human and AI generated text (Crothers et al., 2023; Su et al., 2023; Tulchinskii et al., 2023; Xu et al., 2024). DetectGPT (Mitchell et al., 2023) and its accelerated variants (e.g., Fast-DetectGPT (Bao et al., 2024)) construct detection signals based on perturbation consistency and the geometric properties of log-probability landscapes (Yang et al., 2024a). Binoculars (Hans et al., 2024) further compares probability behaviors between two similar models, achieving strong performance without training an explicit detector. Despite their effectiveness under single-generator assumptions, many of these approaches implicitly adopt a single external language model as a fixed scoring reference. This shared design choice has been observed to limit robustness when generators are heterogeneous or unknown, leading to miscalibration and performance instability across sources.

Multi-Observers and Ensemble-Based Detection. To address cross-generator generalization, several works explore multi-model ensembles (Abburi et al., 2023) or multi-observers frameworks that combine detection signals via averaging (Wu et al., 2025), voting (Kiss and Berend, 2025), or learned weighting schemes (Wu et al., 2024c). MOSAIC (Dubois et al., 2025) adopts an information-theoretic perspective to infer aggregation weights, improving robustness in multi-generator settings. However, most ensemble-based approaches require loading multiple full-scale language models in parallel, causing inference and deployment costs to grow linearly with the number of observers. Furthermore, scaling such systems to cover a large and evolving set of potential generators—especially unknown ones—remains a significant challenge.

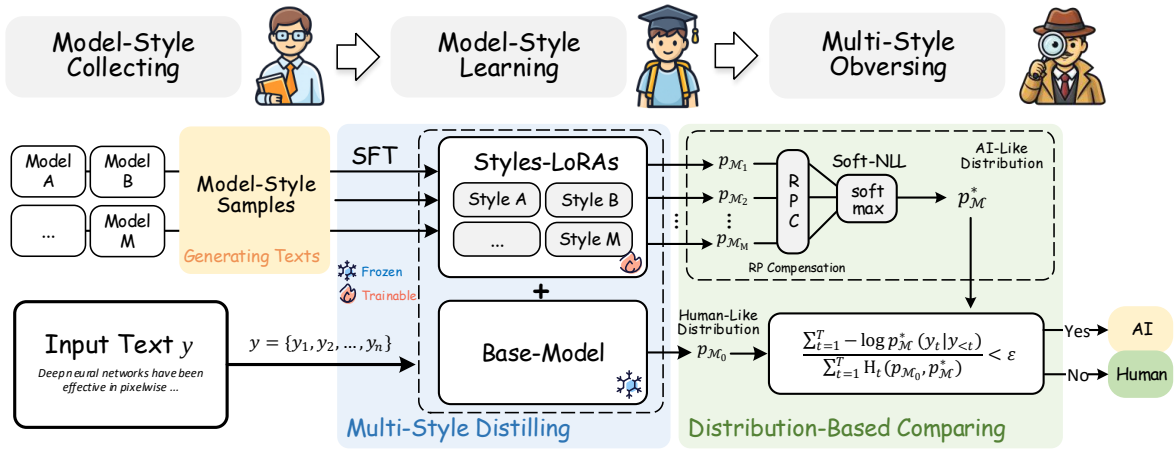

Figure 2: Overview of MSDOS. The framework consists of three stages: (i) Model-Style Collecting from multiple candidate generators, (ii) Model-Style Learning via LoRA-based distillation on a shared backbone, and (iii) Multi-Style Observing by aggregating an AI-side reference and comparing it against a human-side reference for detection.

3 MSDOS Method

Figure 2 provides an overview of MSDOS. We next introduce preliminaries and then detail each stage of the framework.

3.1 Preliminaries

Probabilistic Observation. We represent an input text as a token sequence $y = (y_1, \ldots, y_T)$ over a finite vocabulary $\mathcal{V}$. Given a generation process (language model) $\mathcal{M}$, its conditional distribution at position t is denoted as $p_{\mathcal{M}}(y_t \mid y_{<t})$, where $y_{<t} = (y_1, \ldots, y_{t-1})$. The probability of the entire sequence follows the chain rule:

$$p_{\mathcal{M}}(y) = \prod_{t=1}^{T} p_{\mathcal{M}}(y_t \mid y_{<t}). \tag{1}$$

In subsequent derivations, we primarily work with log-probabilities.

Information-Theoretic Quantities. The negative log-likelihood (NLL) (Cover, 1999) of an observed token y_t under a generation process $\mathcal{M}$ is defined as

$$\mathrm{NLL}_{\mathcal{M}}(y_t \mid y_{<t}) = -\log p_{\mathcal{M}}(y_t \mid y_{<t}), \tag{2}$$

which measures the *surprisal* of the token. Intuitively, smaller NLL indicates that y_t is more predictable under $\mathcal{M}$, while larger NLL suggests a stronger mismatch between the observed token and next-token preference.

Given two conditional distributions $p_{\mathcal{M}_1}(\cdot \mid y_{<t})$ and $p_{\mathcal{M}_2}(\cdot \mid y_{<t})$, we define the conditional cross-entropy as

$$\mathrm{H}_t(p_{\mathcal{M}_1}, p_{\mathcal{M}_2}) \triangleq \mathbb{E}_{x \sim p_{\mathcal{M}_1}}[-\log p_{\mathcal{M}_2}(x \mid y_{<t})], \tag{3}$$

where the expectation is taken over $p_{\mathcal{M}_1}(\cdot \mid y_{<t})$, with the conditioning context $y_{<t}$ shared by both distributions. The conditional cross-entropy measures the expected surprisal assigned by $\mathcal{M}_2$ to tokens sampled from $\mathcal{M}_1$ under the same context.

3.2 Revisiting Binoculars

We revisit the core idea behind perturbation-based zero-shot detection. Binoculars (Hans et al., 2024) typically uses a pair of language models that share the same tokenizer: a *observer* model (e.g., a base model) and a *performer* model (e.g., an instruction model). For an input y, the Binoculars score can be written as the ratio between the observed surprisal of text under performer $p_{\mathcal{M}_2}$ and the expected surprisal under the observer $p_{\mathcal{M}_1}$:

$$\mathrm{Bin}(y) \triangleq \frac{\sum_{t=1}^{T} -\log p_{\mathcal{M}_2}(y_t \mid y_{<t})}{\sum_{t=1}^{N} \mathbb{E}_{x \sim p_{\mathcal{M}_1}}[-\log p_{\mathcal{M}_2}(x \mid y_{<t})]}, \tag{4}$$

where the numerator is the empirical surprisal of the observed tokens under $p_{\mathcal{M}_2}$, and the denominator is the expected surprisal of tokens drawn from $p_{\mathcal{M}_1}$ when evaluated by $p_{\mathcal{M}_2}$. Thus, Binoculars measures how well the text matches the AI-side observer distribution $p_{\mathcal{M}_2}$, using $p_{\mathcal{M}_1}$ as a human-side baseline: well-explained texts yield smaller ratios, while mismatched texts produce larger values.

From our perspective, this formulation compares the observed text against an AI-side observer dis-

tribution $p_{\mathcal{M}_2}$, which serves as the reference in scoring, with $p_{\mathcal{M}_1}$ acting as the human-side expectation baseline. In practice, Binoculars instantiates this pair using a base language model and its instruction-tuned variant: the base model, trained with the standard next-token language-modeling objective on large-scale human-written corpora, serves as a proxy for the natural-text (pretraining) distribution, while instruction tuning shifts the model toward instruction-following behaviors (Wu et al., 2024b). However, this two-model design implicitly assumes a fixed and representative AI-side reference, which becomes fragile under heterogeneous generators, mixed sources, and diverse decoding strategies. To address this limitation, we construct the AI-side reference distribution by aggregating multiple distilled AI-side observers, each capturing a distinct generation style.

3.3 Multi-Style Distillation Observers

Existing zero-shot detectors often rely on a fixed AI-side observer process (e.g., an instruction model as a proxy), which is fragile under *unknown and mixed sources*. To better approximate real-world AI generation behaviors, we propose Multi-Style Distillation Observers, which distill each candidate model's generation style—defined as its systematic preferences over next-token probability distributions—into a shared backbone, rather than training a discriminative detector. Accordingly, rather than assuming a single AI-side process, we explicitly model a set of candidate AI-generation processes $\{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$, where M denotes the number of candidate generators/styles considered and each $\mathcal{M}_m$ corresponds to one distilled observer.

We adopt a shared backbone language model as a unified probability space (without changing the tokenizer), and distill the generation preferences of multiple potential generators into a collection of lightweight style observers, implemented as parameter-efficient adapters (LoRA).

For the m-th target generator, we collect an unlabeled text set $\mathcal{D}_m = \{y^{(i)}\}_{i=1}^{|\mathcal{D}_m|}$ generated under a shared prompt distribution and standard decoding settings. To capture such style-level distributional preferences, we freeze the backbone parameters and train only the adapter parameters θ_m using a standard autoregressive NLL objective:

$$\mathcal{L}(\theta_m) = \mathbb{E}_{y\sim\mathcal{D}_m}\left[-\sum_{t=1}^{T}\log p_{\theta_0+\theta_m}(y_t \mid y_{<t})\right], \tag{5}$$

where $p_{\theta_0+\theta_m}(\cdot \mid y_{<t})$ denotes the next-token distribution of the shared backbone (θ_0) augmented with the m-th adapter (θ_m).

We denote the conditional distribution induced by the m-th observer as

$$p_{\mathcal{M}_m}(\cdot \mid y_{<t}) \triangleq p_{\theta_0+\theta_m}(\cdot \mid y_{<t}), \tag{6}$$

which represents a distinct generation process $\mathcal{M}_m$ within the shared backbone. We denote the backbone (without adapters) as $\mathcal{M}_0$ with distribution $p_{\mathcal{M}_0}(\cdot \mid y_{<t})$, which serves as a proxy for the natural (human-written) text distribution in our framework. Since all observers share the same tokenizer and backbone, their distributions are directly comparable and composable in a unified probability space. As a result, MSDOS avoids the memory and inference overhead of ensembling multiple full language models, and can be efficiently extended to emerging generators by distilling an additional style adapter, without retraining existing observers or modifying the detection objective.

3.4 Decoding-Aware Repetition-Penalty Compensation

In deployment, AI-generated text is commonly produced using decoding strategies such as temperature scaling, greedy or top-k/top-p sampling, and repetition penalty (rp). These decoding strategies induce systematic shifts in token-level probability distributions, violating the assumption that test-time text is sampled from the same distribution used by the detector.

To improve robustness against such decoding-induced distribution shifts, we propose **Repetition-Penalty Compensation (RPC)**. Rather than modifying the detection score itself, RPC directly adjusts the AI-side observer distributions to remain consistent with the actual generation mechanism.

Formally, given an AI-side observer distribution $p_{\mathcal{M}_m}(\cdot \mid y_{<t})$, we apply an rp-consistent transformation operator $\mathcal{T}_{\mathrm{rp}}$, implemented identically to the repetition penalty used in decoding (see Appendix A for the exact formulation), and obtain.

$$p^{\mathrm{rp}}_{\mathcal{M}_m}(\cdot \mid y_{<t}) \triangleq \mathcal{T}_{\mathrm{rp}}(p_{\mathcal{M}_m}(\cdot \mid y_{<t}); \mathrm{rp})\,. \tag{7}$$

When $\mathrm{rp} = 1$, $\mathcal{T}_{\mathrm{rp}}$ reduces to the identity mapping. For $\mathrm{rp} > 1$, RPC explicitly models the suppression of repeated tokens, thereby aligning probability evaluation at detection time with the decoding-time behavior that produced the text.

3.5 Mixture Inference and Scoring

Given the AI-side observer set $\{p_{\mathcal{M}_m}\}_{m=1}^{M}$ constructed in Section 3.3, we aim to build a unified AI-side reference distribution that remains effective under unknown and mixed generators. Following Section 3.4, all observer distributions are first converted into their rp-compensated forms $\{p^{\mathrm{rp}}_{\mathcal{M}_m}\}_{m=1}^{M}$.

To aggregate these observers, we adopt a sentence-level *Soft-NLL* scheme that assigns mixture weights based on how well each compensated observer explains the input text. Specifically, we define the fused AI-side reference distribution as

$$p^{*}_{\mathcal{M}}(\cdot \mid y_{<t}) \triangleq \sum_{m=1}^{M} \mu^{*}_{m}(y)\, p^{\mathrm{rp}}_{\mathcal{M}_m}(\cdot \mid y_{<t}), \quad (8)$$

where $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ denotes the set of candidate AI generation processes. The mixture weights $\mu^{*}(y)$ are obtained via a sentence-level *Soft-NLL* scheme, which assigns higher weights to observers that yield lower sentence-level negative log-likelihood on y. The exact formulation is provided in Appendix B.

Using the fused AI-side reference distribution $p^{*}_{\mathcal{M}}$, we define the final detection score as

$$S(y) \triangleq \frac{\sum_{t=1}^{T} -\log p^{*}_{\mathcal{M}}(y_t \mid y_{<t})}{\sum_{t=1}^{T} \mathrm{H}_t\big(p_{\mathcal{M}_0}, p^{*}_{\mathcal{M}}\big)}, \quad (9)$$

where $\mathrm{H}_t(\mathcal{M}_0, \mathcal{M})$ denotes the conditional cross-entropy defined in Section 3.1. Equation (9) contrasts the observed surprisal of the input under the unified AI-side observer with a baseline expectation induced by the backbone process $\mathcal{M}_0$. Texts exhibiting stronger AI-generation preferences are better explained by $p^{*}_{\mathcal{M}}$ than this baseline, leading to a more separable detection signal. In practice, AI-generated texts tend to yield smaller values.

4 Experiments

4.1 Experimental Setup

Dataset. We conduct experiments on the **RAID** benchmark (Dugan et al., 2024), which is designed for realistic AI-generated text detection with diverse generators, decoding configurations, and adversarial perturbations. In our main setting, we focus on four high-risk domains: *Abstracts*, *News*, *Reddit*, and *Reviews*. For each domain, we construct a balanced split by randomly sampling 3,000 AI-generated texts for distillation and 1,000 human/AI texts for testing.

Distillation Details. To cover heterogeneous generation behaviors, we distill style observers from **GPT-2** (Radford et al., 2019), **MPT** (Team, 2023), and **GPT-4** (OpenAI et al., 2024), representing early autoregressive models, modern open-source instruction-tuned models, and highly aligned proprietary models, respectively. We use **Falcon-rw-1B** (Penedo et al., 2023) as the default backbone, and replicate key results with **Qwen3-0.6B** (Team, 2025) and **Falcon-7B** to validate that our gains are not specific to a particular backbone choice. Unless otherwise specified, we perform one-epoch autoregressive distillation using a base learning rate of 5×10^{-4}, LoRA rank r=8, and α=16. For larger backbones, we reduce the learning rate for training stability (see Appendix C). Each observer is distilled once and reused across all experiments.

Baselines. We compare MSDOS with *RoBERTa (ChatGPT)* (Guo et al., 2023), *Likelihood*, *LogRank* (Solaiman et al., 2019), *LRR* (Su et al., 2023), *DetectGPT* (Mitchell et al., 2023), *Fast-DetectGPT* (Bao et al., 2024), *Binoculars* (Hans et al., 2024), and *MOSAIC* (Dubois et al., 2025). We follow the official code or widely used re-implementations, and adopt the best-reported settings whenever applicable; otherwise we tune on the validation split. Complete configurations are in Appendix D.

Metrics. Following prior work, we report *AUROC* as the primary metric. To quantify detection reliability at low false-positive rates, we additionally report *TPR5%*, the true positive rate at a false positive rate of 5%. All metrics are computed with SCIKIT-LEARN (Pedregosa et al., 2011).

4.2 Main Results

Table 1 presents the main results on the RAID benchmark, where *Mix* denotes a mixed-generator test set combining MPT, GPT-2, and GPT-4. Overall, MSDOS consistently outperforms existing zero-shot detectors across all evaluation settings.

Compared with supervised classifiers, MSDOS achieves comparable or superior performance without relying on labeled training data. More importantly, MSDOS shows clear and stable gains over prior zero-shot methods in both AUROC and TPR5%. While existing approaches typically depend on a single observer model or fixed perturbation assumptions and are sensitive to generator and decoding variations, MSDOS explicitly models diverse generation styles and aggregates them into

Method	MPT		GPT-2		GPT-4		Mix	
	AUROC	TPR5%	AUROC	TPR5%	AUROC	TPR5%	AUROC	TPR5%
RoBERTa-ChatGPT	43.41	17.00	76.88	42.40	71.87	28.00	65.66	31.00
Likelihood(Neo-2.7B)	46.89	30.70	71.24	49.70	83.32	52.70	67.99	45.60
LogRank(Neo-2.7B)	51.24	31.60	74.16	52.30	82.86	55.00	70.41	46.90
LRR(Neo-2.7B)	69.82	48.20	83.18	59.20	79.77	54.00	79.19	55.50
DetectGPT(T5-small/Neo-2.7B)⋄	46.31	15.30	66.15	27.50	74.10	31.10	64.31	29.00
Fast-Detect(GPT-J-6B/Neo-2.7B)	50.04	46.30	84.21	75.10	92.75	76.50	75.98	65.80
Binoculars(Falcon/Falcon-Instruct-7B)	52.73	44.80	78.77	68.75	97.67	91.15	77.20	69.10
MOSAIC(Llama2+, TowerBase+)	68.38	49.70	70.87	44.30	96.62	83.00	79.04	57.40
MSDOS(Falcon-7B)	82.18	69.80	87.83	72.10	**99.41**	97.50	90.55	81.60
MSDOS(Qwen3-0.6B)	90.39	73.20	97.53	88.60	91.25	68.10	93.50	77.10
MSDOS w/o Style(Falcon-rw-1B)	83.63	66.70	96.65	84.20	95.32	80.60	92.75	79.30
MSDOS w/o RPC(Falcon-rw-1B)	78.71	65.00	89.28	76.50	99.05	96.40	89.54	80.60
MSDOS(Falcon-rw-1B)	**95.45**	**90.50**	**98.87**	**95.41**	**99.41**	**97.90**	**97.80**	**93.10**

Table 1: Main results on the RAID benchmark. We report AUROC and TPR5% across different generators. In MOSAIC, "Llama2+" and "TowerBase+" denote ensembles over Llama-2-7B/-Chat and TowerBase-7B/-13B, respectively. MSDOS w/o Style and MSDOS w/o RPC denote ablated variants without multi-style distillation and repetition-penalty compensation, respectively. ⋄ indicates perturbation-based inference is prohibitively slow and is therefore omitted from subsequent experiments. Best results are in **bold** and second-best are underlined.

a unified AI-side observer distribution, resulting in improved robustness under mixed and unseen generators. We also find that the gains are not tied to a specific backbone: MSDOS delivers strong results when instantiated on base models of different scales (e.g., Falcon-rw-1B, Falcon-7B and Qwen3-0.6B in Table 1).

4.3 Analysis and Ablation

Unless otherwise specified, experiments are conducted on RAID samples labeled as *Sampling* with no repetition penalty (RP=1.0), as this setting represents a common and representative generation scenario while avoiding trivial ceiling effects. The effects of different decoding strategies and repetition penalties are analyzed in Section 4.4.

Ablating Style Modeling. We examine the effectiveness of style modeling by comparing three configurations: (i) **No-Style**, which uses only the base model without any style observers; (ii) **Single-Style**, which distills a single observer from one generator; and (iii) **Multi-Style (MSDOS)**, which distills and aggregates observers from multiple generators. As shown in Table 2, **No-Style** consistently yields the weakest performance, indicating that relying solely on the base model's human-side reference is insufficient for detecting diverse AI-generated texts. Introducing a **Single-Style** observer brings clear gains, but exhibits a pronounced *style-alignment* effect: performance improves when the test generator is close to the distillation source yet degrades substantially on out-of-family generators, suggesting miscalibration under cross-style mismatch.

In contrast, **Multi-Style (MSDOS)** achieves the strongest and most stable performance across generators. By aggregating heterogeneous style observers, MSDOS mitigates cross-style mismatch and provides a more representative AI-side reference distribution, thereby improving robustness and generalization.

Fusion Strategy Ablation. We analyze the impact of different style fusion strategies by comparing the adaptive likelihood-based fusion in MSDOS with a simple uniform-weighting baseline. As shown in Table 2, uniform weighting improves over single-style modeling by partially alleviating style bias, but it consistently underperforms adaptive fusion. This indicates that style observers differ substantially in how well they explain a given input; by adjusting mixture weights based on likelihood statistics, MSDOS emphasizes the observers most compatible with the input, which is especially important under mixed-generator and cross-domain settings.

4.4 Robustness and Generalization

We further evaluate the robustness and generalization ability of MSDOS under diverse decoding and penalty configurations provided by RAID, including different decoding labels (greedy vs. sampling) and generation settings with or without repetition

Method	MPT		GPT-2		GPT-4		Mix	
	AUROC	TPR5%	AUROC	TPR5%	AUROC	TPR5%	AUROC	TPR5%
Non-Style	93.59	72.12	98.18	88.75	89.62	58.75	93.96	73.50
MPT-Style	**96.77**	85.25	98.76	92.25	91.42	60.25	95.59	78.25
GPT2-Style	96.57	**85.50**	**99.76**	**99.75**	94.70	77.75	97.07	88.25
GPT4-Style	92.26	63.00	98.26	92.75	**99.68**	**98.75**	97.30	83.67
Multi-Style(Uniform)	95.73	76.60	99.27	98.50	98.06	91.75	98.03	89.75
Multi-Style(Soft-NLL)	95.78	76.75	99.30	98.50	98.54	93.50	**98.23**	**91.00**

Table 2: Ablation study on style modeling strategies. We report AUROC and $\mathrm{TPR5\%}$ on RAID.

	Chat Models				Non-Chat Models				Avg.	
Dec. Strategy	Greedy		Sampling		Greedy		Sampling			
Rep. Penalty	✓	×	✓	×	✓	×	✓	×		
Method	AUROC	AUROC	AUROC	AUROC	AUROC	AUROC	AUROC	AUROC	AUROC	TPR5%
Likelihood	92.96	97.98	72.58	96.08	65.58	99.97	9.48	74.02	76.08	59.16
log-rank	93.91	98.60	74.36	96.83	71.42	99.99	12.08	78.01	78.15	60.22
LLR	94.98	98.77	77.76	96.54	92.17	99.88	30.43	83.81	84.29	64.94
Fast-Detect	93.90	99.85	71.12	99.21	62.27	99.98	11.50	96.60	79.30	71.81
Binoculars	97.84	100.00	81.32	99.94	70.03	100.00	8.90	94.98	81.63	74.72
MOSAIC	99.15	99.92	87.41	99.14	86.89	95.81	9.95	89.18	83.43	71.84
MSDOS	**99.82**	**100.00**	**99.07**	**99.70**	**99.97**	**99.99**	**93.68**	**93.41**	**98.21**	**93.58**

Table 3: Detection performance under different decoding strategies and repetition penalty (RP) settings. We report AUROC for each setting, and the averaged AUROC and TPR5% in the last two columns. Chat models include llama-chat, mistral-chat, and mpt-chat, while non-chat models include mistral, mpt, and gpt2.

penalty. Such decoding variations substantially alter the generation distribution and pose well-known challenges to existing zero-shot detectors.

Impact of Generation Decoding Strategies. Prior work has shown that many zero-shot detection methods are highly sensitive to decoding strategies, as their underlying assumptions often fail to hold under practical generation settings. We therefore evaluate MSDOS under different decoding strategies (greedy vs. sampling) on the RAID benchmark. As reported in Table 3, MSDOS maintains relatively stable performance across decoding configurations, exhibiting substantially smaller performance fluctuations than baseline methods. This indicates that distilling style observers directly from real generated texts improves tolerance to *decoding-induced* distribution variations, leading to more robust detection across diverse generation behaviors.

Effect of Repetition Penalty in Generation. Repetition penalty is widely adopted in practice but can significantly alter repetition patterns and token-level probability profiles, introducing a pronounced form of decoding-induced distribution shift. As shown in Table 3, most baseline detectors degrade noticeably under repetition-penalized generation, highlighting their sensitivity to such shifts. In contrast, MSDOS remains stable across decoding strategies and repetition-penalty settings. Moreover, the ablation results in Table 1 show that removing RPC leads to a clear performance drop on RP-enabled samples, confirming the importance of RPC for improving robustness to repetition-penalty-induced distribution shifts.

Ablation on RPC Strength. We analyze the sensitivity of MSDOS to the RPC strength rp, a detection-time hyperparameter used to offset repetition-penalty-induced shifts in the AI-side observers. Varying $\mathrm{rp} \in [1.0, 1.3]$ shows that performance remains stable within a reasonable range but exhibits a clear operating region (Figure 4): small rp under-compensates, while large rp introduces noise and degrades stability. Moderate rp values strike a balance between the uncompensated and over-compensated regimes, leading to more robust detection performance. We choose $\mathrm{rp} = 1.1$ as the default, which achieves the best or near-best performance across settings.

5 Reliability in the Wild

5.1 Generalization to Unseen Generators and Domains

We evaluate the generalization of MSDOS under two realistic distribution shifts: *unseen generators* and *unseen domains*. For unseen generators

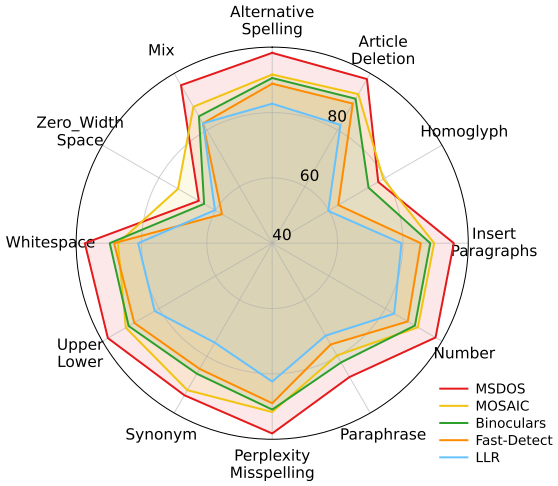

Figure 3: Radar chart illustrating the impact of 11 types of attack on detection performance (AUROC) on RAID.

(Table 7), the test-time generators are excluded from style distillation; MSDOS remains consistently strong, whereas most baselines rely heavily on specific generator families and degrade substantially when transferred to unseen models. For unseen domains (Table 8), we train the detector on a subset of text domains and evaluate it on entirely held-out ones; MSDOS shows smaller performance variance across domains and remains effective even when the target domains differ markedly in style. Overall, these results indicate that multi-style distillation with distribution-level aggregation improves robustness to both generator and domain shifts, reducing reliance on any single model family or domain-specific cues.

5.2 Robustness to Attacks

We further assess the robustness of MSDOS under various adversarial attack settings provided by the RAID benchmark. As shown in Figure 3, MSDOS achieves the best performance on most attack types, indicating strong robustness to text perturbations and noise injection. Notably, the largest performance drops are observed for character-level attacks such as *Zero_Width_Space* and *Homoglyph*, where MSDOS remains competitive but is less dominant than in other settings. Overall, these results suggest that the information fusion across multiple style observers helps mitigate the impact of diverse adversarial manipulations by evaluating input texts from complementary distributional perspectives.

Method	Time (s)↓	GPU Memory (MB)↓
LLR	**153**	6,560
Fast-Detect	216	18,460
Binoculars	282	29,130
MOSAIC	1583	37,640
MSDOS	192	**4,560**

Table 4: Detection computational cost on RAID.

5.3 Additional Benchmark Validation

We additionally evaluate MSDOS on the public **HART** (Bao et al., 2025) benchmark to verify that its effectiveness does not hinge on a particular dataset construction or distributional assumption. HART consists of 3-level tasks with increasing difficulty, including human rewriting and AI polishing. As shown in Table 9, MSDOS achieves the best performance on most tasks and levels, while ranking near second on *Writing* in level 1. This behavior may be attributed to the fact that machine edits on human-written content can preserve much of the original distribution, reducing the separability.

5.4 Computational Cost and Memory

As shown in Table 4, MSDOS achieves the smallest GPU memory footprint during detection, since it avoids loading and running multiple full-scale language models as in multi-model or ensemble-based baselines. It is also faster than most compared methods in our end-to-end detection setting, which reduces deployment-time overhead in practice. During distillation, MSDOS supports both serial and parallel execution, allowing flexible trade-offs between time and hardware resources, with details reported in Appendix C.

6 Conclusion

We addressed the challenge of AI-generated text detection in open-world settings with unknown and mixed generators. We proposed MSDOS, a multi-style distillation framework that aggregates heterogeneous generation styles into a unified AI-side observer distribution without requiring labeled data. Extensive experiments show that MSDOS consistently outperforms prior zero-shot detectors under generator heterogeneity and decoding-induced distribution shifts. Our ablation studies further validate the robustness of MSDOS across backbone choices and decoding configurations. This work highlights the importance of distribution-level style modeling for reliable detection in realistic scenarios.

Limitations

MSDOS does not require labeled human–AI pairs or access to target generators, but it still relies on a non-trivial amount of unlabeled AI-generated text to distill style observers, and the coverage of generators/decoding settings in the distillation corpus can affect robustness in the wild. Our observers are trained via autoregressive NLL distillation, which captures distributional tendencies in next-token preferences rather than recovering exact generator probabilities, and may miss fine-grained probabilistic structures. Finally, style fidelity is bounded by the shared backbone's capacity and tokenizer/vocabulary; distilling from substantially larger generators into a compact backbone can introduce an information bottleneck, leaving the effect of backbone scale and tokenizer design an open direction for future study.

References

Harika Abburi, Kalyani Roy, Michael Suesserman, Nirmala Pudota, Balaji Veeramani, Edward Bowen, and Sanmitra Bhattacharya. 2023. A simple yet efficient ensemble approach for AI-generated text detection. In *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 413–421, Singapore. Association for Computational Linguistics.

Duarte M. Alves, José Pombal, Nuno M. Guerreiro, Pedro H. Martins, João Alves, Amin Farajian, Ben Peters, Ricardo Rei, Patrick Fernandes, Sweta Agrawal, Pierre Colombo, José G. C. de Souza, and André F. T. Martins. 2024. Tower: An open multilingual large language model for translation-related tasks. *Preprint*, arXiv:2402.17733.

Jinze Bai and 1 others. 2023. Qwen technical report. Technical report, Alibaba Group.

Guangsheng Bao, Lihua Rong, Yanbin Zhao, Qiji Zhou, and Yue Zhang. 2025. Decoupling content and expression: Two-dimensional detection of ai-generated text. *Preprint*, arXiv:2503.00258.

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*.

Yutian Chen, Hao Kang, Vivian Zhai, Liangze Li, Rita Singh, and Bhiksha Raj. 2023. Gpt-sentinel: Distinguishing human and chatgpt generated content. *Preprint*, arXiv:2305.07969.

Thomas M Cover. 1999. *Elements of information theory*. John Wiley & Sons.

Evan N. Crothers, Nathalie Japkowicz, and Herna L. Viktor. 2023. Machine-generated text: A comprehensive survey of threat models and detection methods. *IEEE Access*, 11:70977–71002.

Matthieu Dubois, François Yvon, and Pablo Piantanida. 2025. MOSAIC: Multiple observers spotting AI content. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 24230–24247, Vienna, Austria. Association for Computational Linguistics.

Liam Dugan, Alyssa Hwang, Filip Trhlík, Andrew Zhu, Josh Magnus Ludan, Hainiu Xu, Daphne Ippolito, and Chris Callison-Burch. 2024. RAID: A shared benchmark for robust evaluation of machine-generated text detectors. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12463–12492, Bangkok, Thailand. Association for Computational Linguistics.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, and 1 others. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

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.

Soumya Suvra Ghosal, Souradip Chakraborty, Jonas Geiping, Furong Huang, Dinesh Manocha, and Amrit Singh Bedi. 2023. Towards possibilities & impossibilities of ai-generated text detection: A survey. *arXiv preprint arXiv:2310.15264*.

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 *International Conference on Machine Learning*, pages 17519–17537. PMLR.

Wayne Holmes, Fengchun Miao, and 1 others. 2023. *Guidance for generative AI in education and research*. Unesco Publishing.

Mihaly Kiss and Gábor Berend. 2025. SzegedAI at GenAI detection task 1: Beyond binary - soft-voting multi-class classification for binary machine-generated text detection across diverse language models. In *Proceedings of the 1stWorkshop on GenAI*

Content Detection (GenAIDetect), pages 166–172, Abu Dhabi, UAE. International Conference on Computational Linguistics.

Kristian Kuznetsov, Eduard Tulchinskii, Laida Kushnareva, German Magai, Serguei Barannikov, Sergey Nikolenko, and Irina Piontkovskaya. 2024. Robust AI-generated text detection by restricted embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 17036–17055, Miami, Florida, USA. 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, Bangkok, Thailand. Association for Computational Linguistics.

Yikang Liu, Ziyin Zhang, Wanyang Zhang, Shisen Yue, Xiaojing Zhao, Xinyuan Cheng, Yiwen Zhang, and Hai Hu. 2023. Argugpt: evaluating, understanding and identifying argumentative essays generated by gpt models. *Preprint*, arXiv:2304.07666.

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*, volume 202 of *Proceedings of Machine Learning Research*, pages 24950–24962. PMLR.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, and 1 others. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Guilherme Penedo, Quentin Malartic, Daniel Hesslow, Ruxandra Cojocaru, Alessandro Cappelli, Hamza Alobeidli, Baptiste Pannier, Ebtesam Almazrouei, and Julien Launay. 2023. The RefinedWeb dataset for Falcon LLM: outperforming curated corpora with web data, and web data only. *arXiv preprint arXiv:2306.01116*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Aquia Richburg, Calvin Bao, and Marine Carpuat. 2024. Automatic authorship analysis in human-AI collaborative writing. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 1845–1855, Torino, Italia. ELRA and ICCL.

Areg Mikael Sarvazyan, José Ángel González, Paolo Rosso, and Marc Franco-Salvador. 2023. Supervised machine-generated text detectors: Family and scale matters. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, pages 121–132, Cham. Springer Nature Switzerland.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, Gretchen Krueger, Jong Wook Kim, Sarah Kreps, Miles McCain, Alex Newhouse, Jason Blazakis, Kris McGuffie, and Jasmine Wang. 2019. Release strategies and the social impacts of language models. *Preprint*, arXiv:1908.09203.

Jinyan Su, Terry Zhuo, Di Wang, and Preslav Nakov. 2023. DetectLLM: Leveraging log rank information for zero-shot detection of machine-generated text. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12395–12412, Singapore. Association for Computational Linguistics.

Zhen Sun, Zongmin Zhang, Xinyue Shen, Ziyi Zhang, Yule Liu, Michael Backes, Yang Zhang, and Xinlei He. 2025. Are we in the AI-generated text world already? quantifying and monitoring AIGT on social media. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 22975–23005, Vienna, Austria. Association for Computational Linguistics.

MosaicML NLP Team. 2023. Introducing mpt-7b: A new standard for open-source, ly usable llms.

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Hugo Touvron and 1 others. 2023. Llama: Open and efficient foundation language models. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Eduard Tulchinskii, Kristian Kuznetsov, Laida Kushnareva, Daniil Cherniavskii, Sergey Nikolenko, Evgeny Burnaev, Serguei Barannikov, and Irina Piontkovskaya. 2023. Intrinsic dimension estimation for robust detection of ai-generated texts. *Advances in Neural Information Processing Systems*, 36:39257–39276.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, Thomas Arnold, Alham Fikri Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024. M4: Multi-generator, multi-domain, and multilingual black-box machine-generated text detection. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1369–1407, St. Julian's, Malta. Association for Computational Linguistics.

Zecong Wang, Jiaxi Cheng, Chen Cui, and Chenhao Yu. 2023. Implementing bert and fine-tuned roberta to detect ai generated news by chatgpt. *Preprint*, arXiv:2306.07401.

Junchao Wu, Runzhe Zhan, Derek F. Wong, Shu Yang, Xinyi Yang, Yulin Yuan, and Lidia S. Chao. 2024a. Detectrl: Benchmarking llm-generated text detection in real-world scenarios. In *Advances in Neural Information Processing Systems*, volume 37, pages 100369–100401. Curran Associates, Inc.

Junxi Wu, Jinpeng Wang, Zheng Liu, Bin Chen, Dongjian Hu, Hao Wu, and Shu-Tao Xia. 2025. MoSEs: Uncertainty-aware AI-generated text detection via mixture of stylistics experts with conditional thresholds. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5797–5816, Suzhou, China. Association for Computational Linguistics.

Xuansheng Wu, Wenlin Yao, Jianshu Chen, Xiaoman Pan, Xiaoyang Wang, Ninghao Liu, and Dong Yu. 2024b. From language modeling to instruction following: Understanding the behavior shift in LLMs after instruction tuning. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2341–2369, Mexico City, Mexico. Association for Computational Linguistics.

Youlin Wu, Kaichun Wang, Kai Ma, Liang Yang, and Hongfei Lin. 2024c. Werkzeug at SemEval-2024 task 8: LLM-generated text detection via gated mixture-of-experts fine-tuning. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 547–552, Mexico City, Mexico. Association for Computational Linguistics.

Yang Xu, Yu Wang, Hao An, Zhichen Liu, and Yongyuan Li. 2024. Detecting subtle differences between human and model languages using spectrum of relative likelihood. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 10108–10121, Miami, Florida, USA. Association for Computational Linguistics.

Xianjun Yang, Wei Cheng, Yue Wu, Linda Ruth Petzold, William Yang Wang, and Haifeng Chen. 2024a. DNA-GPT: Divergent n-gram analysis for training-free detection of GPT-generated text. In *The Twelfth International Conference on Learning Representations*.

Xianjun Yang, Liangming Pan, Xuandong Zhao, Haifeng Chen, Linda Ruth Petzold, William Yang Wang, and Wei Cheng. 2024b. A survey on detection of LLMs-generated content. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 9786–9805, Miami, Florida, USA. Association for Computational Linguistics.

Qihui Zhang, Chujie Gao, Dongping Chen, Yue Huang, Yixin Huang, Zhenyang Sun, Shilin Zhang, Weiye Li, Zhengyan Fu, Yao Wan, and Lichao Sun. 2024. LLM-as-a-coauthor: Can mixed human-written and machine-generated text be detected? In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 409–436, Mexico City, Mexico. Association for Computational Linguistics.

Liu Zhuang, Lin Wayne, Shi Ya, and Zhao Jun. 2021. A robustly optimized BERT pre-training approach with post-training. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 1218–1227, Huhhot, China. Chinese Information Processing Society of China.

A Repetition-Penalty-Consistent Transformation

Repetition penalty. Repetition penalty (RP) is a commonly used decoding heuristic to discourage the model from repeatedly generating tokens that have already appeared in the prefix. In this appendix, we provide the exact formulation of the rp-consistent transformation operator $\mathcal{T}_{\mathrm{rp}}$ used in our method.

Let $z(\cdot \mid y_{<t}) \in \mathbb{R}^{|\mathcal{V}|}$ denote the pre-softmax logits of an AI-side observer model given the prefix $y_{<t}$, where $\mathcal{V}$ is the vocabulary. Given a repetition penalty parameter $\mathrm{rp} \geq 1$, we first adjust the logits for tokens that have appeared in the prefix. Specifically, for any token $v \in \mathcal{V}$, we define the transformed logit $\tilde{z}(v \mid y_{<t})$ as

$$\tilde{z}(v \mid y_{<t}) = \begin{cases} g_{\mathrm{rp}}\big(z(v \mid y_{<t})\big), & v \in \mathcal{V}(y_{<t}), \\ z(v \mid y_{<t}), & v \notin \mathcal{V}(y_{<t}), \end{cases} \tag{10}$$

$$g_{\mathrm{rp}}(x) = \begin{cases} x/\mathrm{rp}, & x > 0, \\ x \cdot \mathrm{rp}, & x \leq 0. \end{cases} \tag{11}$$

where $\mathcal{V}(y_{<t})$ denotes the set of tokens that occur at least once in the prefix $y_{<t}$. This piecewise scaling follows the standard repetition penalty implementation used during decoding, ensuring that repeated tokens are suppressed regardless of the sign of their logits.

After applying the logit-level transformation, we obtain the rp-consistent probability distribution by normalizing with the softmax operator:

$$p^{\mathrm{rp}}(v \mid y_{<t}) = \frac{\exp(\tilde{z}(v \mid y_{<t}))}{\sum_{u \in \mathcal{V}} \exp(\tilde{z}(u \mid y_{<t}))}. \tag{12}$$

Accordingly, the rp-consistent transformation operator $\mathcal{T}_{\mathrm{rp}}$ maps the original observer distribution $p(\cdot \mid y_{<t})$ (equivalently, its logits) to the adjusted distribution $p^{\mathrm{rp}}(\cdot \mid y_{<t})$.

When $\mathrm{rp} = 1$, Eq. (10) reduces to the identity transformation, and $\mathcal{T}_{\mathrm{rp}}$ leaves the original distribution unchanged. For $\mathrm{rp} > 1$, the transformation explicitly down-weights tokens that have already appeared in the prefix, thereby aligning probability evaluation at detection time with the decoding-time behavior that generated the text.

B Soft-NLL Mixture Aggregation

Soft-NLL Aggregation. Given a set of rp-consistent AI-side observer distributions $\{p^{\mathrm{rp}}_{\mathcal{M}_m}(\cdot \mid y_{<t})\}_{m=1}^{M}$, we aim to construct a single aggregated AI observer distribution that remains effective under unknown or mixed generators. To this end, we adopt a sentence-level *Soft-NLL* aggregation scheme, which assigns mixture weights based on how well each observer explains the input text.

Let $y = (y_1, \ldots, y_T)$ denote the input sequence. For each observer $\mathcal{M}_m$, we define its (rp-consistent) sentence-level negative log-likelihood (NLL) as

$$\mathrm{NLL}_m(y) = -\frac{1}{T} \sum_{t=1}^{T} \log p^{\mathrm{rp}}_{\mathcal{M}_m}(y_t \mid y_{<t}), \tag{13}$$

where length normalization is applied to avoid bias toward shorter sequences.

We then convert the NLL scores into soft mixture weights via a temperature-controlled softmax:

$$\mu_m = \frac{\exp(-\mathrm{NLL}_m(y)/\tau)}{\sum_{j=1}^{M} \exp(-\mathrm{NLL}_j(y)/\tau)}, \sum_{m=1}^{M} \mu_m = 1, \tag{14}$$

where $\tau > 0$ is a temperature parameter controlling the sharpness of the weighting. Smaller τ emphasizes the best-matching observer, while larger τ yields a more uniform mixture. In all experiments, we set $\tau = 0.5$ unless otherwise stated. Soft-NLL can be viewed as a lightweight, likelihood-based alternative to information-theoretic aggregation schemes, trading theoretical optimality for simplicity and computational efficiency.

Sentence-Level vs. Token-Level Aggregation. We perform aggregation at the sentence level rather than the token level. While token-level weighting allows fine-grained adaptation, it is also more sensitive to local fluctuations caused by a small number of tokens, which may be affected by noise, rare events, or decoding artifacts. In contrast, sentence-level aggregation integrates evidence across the entire sequence, providing a more stable estimate of how well each observer explains the input text. This design reduces the risk of spurious dominance from a few high- or low-probability tokens and leads to more robust mixture weights under unknown or mixed generators.

C Experimental Setup Details

All experiments are conducted on a server equipped with a single NVIDIA Tesla A6000 GPU. We report detailed training configurations here to facilitate reproducibility and future extensions.

Model	Parameters	Learning Rate	Trainable
Qwen3	0.6B	5e-4	0.4%
Falcon-rw	1.0B	5e-4	0.47%
Falcon	7B	5e-5	0.23%

Table 5: Training hyperparameters for different base models used in our experiments.

Setting	Time (s)	GPU Memory (MB)
MSDOS (Serial)	1,155	16,780
MSDOS (Parallel)	372	16,780 $\times$ 3

Table 6: **Distillation-stage efficiency of MSDOS.** Parallel distillation distributes style observers across GPUs to reduce wall-clock time, while serial distillation provides a memory-efficient alternative using a single GPU. "$\times$3" denotes per-GPU memory usage when three GPUs are used.

Distillation Configuration. Unless otherwise specified, models are trained with a batch size of 5 for a single epoch. We use the AdamW optimizer (`adamw_torch`) with a learning rate shown in Table 5, weight decay of 0.01, and a warmup ratio of 0.03. For Falcon-7B, we use a smaller learning rate 5e-5 to avoid training instability at larger model scale. A cosine learning rate scheduler is adopted throughout training.

LoRA Settings. We employ Low-Rank Adaptation (LoRA) for parameter-efficient fine-tuning. Specifically, we set the LoRA rank to $r = 8$, the scaling factor to $\alpha = 16$, and the dropout rate to 0.05. LoRA adapters are applied only to the attention and MLP layers, while all other parameters of the base model are kept frozen.

Distillation Cost and Memory. We report the wall-clock time and peak GPU memory usage of the distillation stage under both serial and parallel execution settings, as summarized in Table 6. The distillation process is performed offline as a one-time cost prior to deployment. Serial distillation trains style observers sequentially on a single GPU, providing a memory-efficient option when hardware resources are limited. Parallel distillation distributes observers across multiple GPUs to reduce overall training time, at the cost of increased aggregate GPU memory usage.

D Baseline Details

Below is a detailed introduction to the setup of the comparative experiment.

- **RoBERTa-ChatGPT** (Guo et al., 2023): A supervised detector refers to a RoBERTa-base model (Zhuang et al., 2021) that has been fine-tuned on the HC3 (Guo et al., 2023) dataset. We choose this model as a representative baseline for supervised detectors.

- **Likelihood and Logrank**: Two simple established zero-shot techniques using per-token negative log-likelihood (equivalent to NLL), and the average logarithmic ranking of tokens in a probability distribution. We uses GPT-neo-2.7B (Gao et al., 2020) as observer model following Bao et al. (2024) experiments.

- **LRR** (Su et al., 2023): A classic detector based on NLL, calculated by dividing the NLL by the log-rank of a scoring model. We uses GPT-neo-2.7B as observer model following Bao et al. (2024) experiments.

- **DetectGPT** (Mitchell et al., 2023): A zero-shot likelihood-ratio detector that measures the discrepancy between an input and its T5-based perturbations under a scoring LM. Following the experimental setup in Bao et al. (2024), we adopt T5-small (Raffel et al., 2020) to generate perturbations and GPT-Neo-2.7B as the scoring model.

- **Fast-DetectGPT** (Bao et al., 2024): A fast and improved version of DetectGPT that utilizes computational dual models to calculate probability distributions and quickly compute perturbed scores. Following the experimental setup in Bao et al. (2024), we adopt GPT-J-6B (Wang and Komatsuzaki, 2021) as the reference model and GPT-Neo-2.7B as the scoring model.

- **Binoculars** (Hans et al., 2024): Another NLL-based detection method operates by dividing the NLL of the performer model by the cross entropy between the observed model and the performer model. In our experiment, we adhered to the original setup, where Falcon-7B-instruction (Penedo et al., 2023) served as the performer model and Falcon-7B served as the observer model.

- **MOSAIC** (Dubois et al., 2025): A multi-observer zero-shot detector that aggregates signals from multiple reference models using an information-theoretic weighting scheme,

aiming to improve robustness across diverse generators. In our experiments, we strictly follow the original MOSAIC setup: TowerBase-13B (Alves et al., 2024) is used as the reference model, while TowerBase-13B, TowerBase-7B, Llama-2-7B-chat, and Llama-2-7B (Touvron et al., 2023) are employed as scoring models. Due to computational constraints, all models are evaluated using 8-bit inference. We observe that the resulting performance deviation is within 0.3%, which is negligible and does not affect the comparative conclusions.

E Detailed Experimental Results

Due to space limitations, we have compiled Table 10 with more experimental details to verify the completeness of the experiments.

F More Discussion

Base model choice for style distillation. In our experiments, we found that the strongest or most recent models are not necessarily the best backbones for MSDOS. Instead, base language models that better reflect the natural-text (human-written) pretraining distribution, as well as smaller models with lightweight adaptation, often work better in practice. One possible reason is that such backbones can more easily absorb diverse generation styles during distillation, whereas heavily optimized or strongly specialized models may exhibit more pronounced inherent preferences, making them harder to adapt toward other styles. This observation is also consistent with the common practice in distillation and student training, where the student is typically trained from scratch or from a neutral initialization rather than starting from a highly specialized model.

How many styles are needed? Prior work suggests that models within the same family tend to exhibit similar generation patterns, implying that a small set of representative styles may cover a large portion of commonly used generators. This motivates our design choice to distill a limited number of diverse style observers instead of enumerating all possible generators. In future work, we will evaluate a broader set of candidate styles, identify robust combinations for realistic mixed-source settings, and explore alternative configurations for constructing style observer sets.

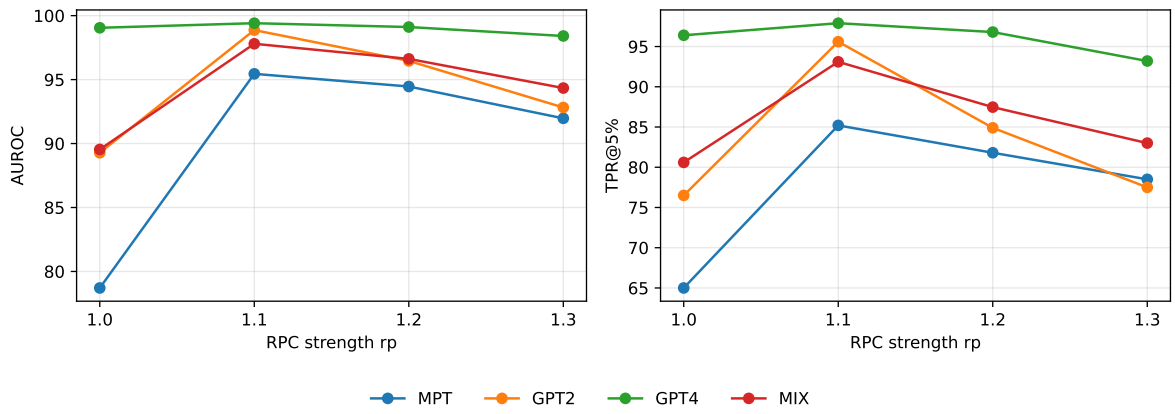

Figure 4: Effect of the repetition-penalty compensation strength rp on detection performance. Performance is stable within a moderate range, with degradation observed for overly small or large rp.

Method	ChatGPT		Cohere-Chat		Cohere		Llama-2-Chat		Mistral-Chat		Mistral		MPT-Chat	
	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.
Likelihood	89.85	43.50	81.61	32.75	76.50	20.75	88.89	50.50	85.11	37.50	63.33	28.25	70.56	20.50
Logrank	90.05	45.00	82.22	36.50	76.54	21.50	89.75	56.00	86.62	41.75	65.51	28.50	72.40	23.25
LLR	89.01	56.50	82.31	48.25	74.74	26.25	90.97	67.25	89.04	57.00	72.40	37.00	76.54	36.25
Fast-Detect	99.60	98.25	95.50	86.50	96.21	84.50	98.07	95.25	93.27	80.50	68.70	57.25	72.74	60.75
Binoculars	99.84	**99.50**	97.78	93.75	98.58	94.75	98.58	97.75	97.15	91.50	69.69	59.00	80.10	74.75
MOSAIC	97.54	84.25	**98.54**	**95.75**	97.91	93.00	99.14	96.25	96.62	85.00	70.86	56.25	86.39	76.25
MSDOS	**99.73**	99.00	97.80	91.50	**99.91**	**99.75**	**99.63**	**98.75**	**99.63**	**98.7**	**96.00**	**82.75**	**99.48**	**98.00**

Table 7: Detection performance in unseen generators on RAID. We report AUROC and TPR5% for 7 unseen generators.

Method	Books		Poetry		Recipes		Wiki		Avg.	
	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.
Likelihood	86.93	76.00	80.55	35.00	80.54	72.25	78.80	64.75	81.71	62.00
Logrank	87.67	77.50	82.03	39.25	80.58	72.50	80.32	66.00	82.65	63.81
LLR	90.45	75.25	84.87	49.50	80.07	65.00	84.64	64.50	85.01	63.56
Fast-Detect	87.77	80.75	84.94	72.50	84.65	78.75	85.65	81.75	85.75	78.44
Binoculars	89.24	87.00	90.17	85.00	85.42	82.75	86.20	80.00	87.76	83.69
MOSAIC	91.30	85.50	88.85	79.00	85.60	83.00	88.06	75.75	88.45	80.81
MSDOS	**99.49**	**97.50**	**99.17**	**97.00**	**98.54**	**95.50**	**98.58**	**95.75**	**98.95**	**96.44**

Table 8: Detection performance in unseen domains on RAID. We report AUROC and TPR5% for each domain, and the averaged in the last two columns.

Method	Level 3				Level 2				Level 1			
	Arxiv	Essay	News	Writing	Arxiv	Essay	News	Writing	Arxiv	Essay	News	Writing
Likelihood	84.88	91.93	68.72	76.06	29.42	60.42	26.20	36.76	59.61	85.35	52.54	61.20
Logrank	86.03	90.76	70.67	74.73	28.62	54.33	26.38	33.92	56.99	82.53	50.54	58.79
LLR	87.32	82.19	78.33	68.33	28.15	36.56	32.90	27.57	47.29	67.31	46.46	50.32
Fast-Detect	88.79	87.13	84.34	79.19	51.47	55.76	46.12	43.36	70.87	79.76	65.32	64.59
Binoculars	91.13	97.06	89.91	87.74	59.85	77.21	56.82	58.80	75.36	90.51	67.08	**73.31**
MOSAIC	95.68	98.17	92.35	90.84	78.00	85.05	65.87	62.08	80.97	94.13	73.31	73.09
MSDOS	**98.93**	**99.98**	**99.26**	**93.74**	**90.91**	**99.51**	**87.20**	**65.00**	**86.92**	**99.77**	**84.76**	65.42

Table 9: We report the detection performance (AUROC) on three level tasks on HART. TPR5% results show consistent trends and are omitted for brevity.

	Chat Models								Non-Chat Models							
Dec. Strategy	Greedy				Sampling				Greedy				Sampling			
Rep. Penalty	✓		×		✓		×		✓		×		✓		×	
Method	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.	AUR.	TPR.
Likelihood	92.96	80.00	97.98	94.50	72.58	39.50	96.08	88.25	65.58	56.25	99.97	**100.00**	9.48	0.25	74.02	14.50
log-rank	93.91	81.50	98.60	94.50	74.36	41.50	96.83	88.75	71.42	58.00	99.99	**100.00**	12.08	0.50	78.01	17.00
LLR	94.98	83.00	98.77	94.50	77.76	43.50	96.54	86.50	92.17	79.75	99.88	99.50	30.43	1.25	83.81	31.50
Fast-Detect	93.90	83.75	99.85	99.25	71.12	50.25	99.21	97.50	62.27	56.25	99.98	**100.00**	11.50	0.50	**96.60**	**87.00**
Binoculars	97.84	95.75	**100.00**	**100.00**	81.32	67.75	99.94	99.75	70.03	60.75	**100.00**	**100.00**	8.90	1.25	94.98	72.50
MOSAIC	99.15	98.00	99.92	**100.00**	87.41	67.25	99.14	96.50	86.89	81.25	95.81	72.50	9.95	1.50	89.18	57.75
MSDOS	**99.82**	**99.75**	**100.00**	**100.00**	**99.07**	**98.50**	**99.70**	**99.50**	**99.97**	**99.75**	99.99	**100.00**	**93.68**	**84.50**	93.41	66.62

Table 10: Detailed Detection performance under different decoding strategies and repetition penalty (RP) settings.