DetectAnyLLM: Towards Generalizable and Robust Detection of Machine-Generated Text Across Domains and Models

Jiachen Fu
VCIP, CS, Nankai University
Tianjin, China
fujiachen2005@gmail.com

Chun-Le Guo*
VCIP, CS, Nankai University
Tianjin, China
NKIARI
Shenzhen Futian, China
guochunle@nankai.edu.cn

Chongyi Li†
VCIP, CS, Nankai University
Tianjin, China
NKIARI
Shenzhen Futian, China
lichongyi@nankai.edu.cn

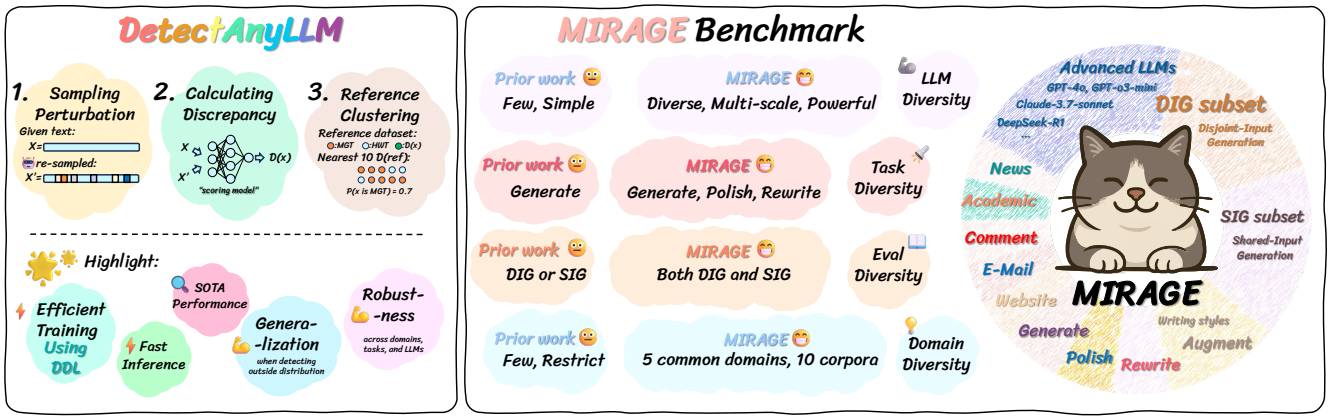

Figure 1: Left: Our DetectAnyLLM achieves high efficiency, strong robustness, and impressive generalization through a three-step process: *sampling perturbation*, *calculating discrepancy*, and *reference clustering*. Right: Our MIRAGE benchmark emphasizes diversity across domains, tasks, evaluation scenarios, and source LLMs, enabling comprehensive and robust evaluation.

Abstract

The rapid advancement of large language models (LLMs) has drawn urgent attention to the task of machine-generated text detection (MGTD). However, existing approaches struggle in complex real-world scenarios: zero-shot detectors rely heavily on scoring model's output distribution while training-based detectors are often constrained by overfitting to the training data, limiting generalization. We found that the performance bottleneck of training-based detectors stems from the misalignment between training objective and task needs. To address this, we propose **Direct Discrepancy Learning (DDL)**, a novel optimization strategy that directly optimizes the detector with task-oriented knowledge. DDL enables the detector to better capture the core semantics of the detection task, thereby enhancing both robustness and generalization. Built upon this, we introduce **DetectAnyLLM**, a unified detection framework that achieves state-of-the-art MGTD performance across diverse LLMs. To ensure a reliable evaluation, we construct **MIRAGE**, the most diverse multi-task MGTD benchmark. MIRAGE samples human-written texts from 10 corpora across 5 text-domains, which are then re-generated or revised using 17 cutting-edge LLMs, covering a wide spectrum of proprietary models and textual styles. Extensive experiments on MIRAGE reveal the limitations of existing methods in complex environment. In contrast, DetectAnyLLM consistently outperforms them, achieving over a 70% performance improvement under the same training data and base scoring model, underscoring the effectiveness of our DDL. Project page: https://fjc2005.github.io/detectanyllm.

CCS Concepts

• **Computing methodologies** → **Artificial intelligence**; **Natural language processing**; • **Security and privacy**;

Keywords

Machine-Generated Text Detection, AI-Text Detection, AI Safety

ACM Reference Format:

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 (MM '25), October 27–31, 2025, Dublin, Ireland.* ACM, New York, NY, USA, 23 pages. https://doi.org/10.1145/3746027.3754862

*Corresponding Author.
†Project Lead.

MM '25, Dublin, Ireland

ACM ISBN 979-8-4007-2035-2/2025/10
https://doi.org/10.1145/3746027.3754862

1 Introduction

Advanced *Large Language Models (LLMs)* [2, 18, 24, 26, 31, 48] can easily generate text nearly indistinguishable from human writing [10, 45]. If misused, it could pose serious risks to society [52]. In response to such concern, the task of *Machine-Generated Text Detection (MGTD)* has emerged [1, 14, 25, 27, 29, 44]. MGTD is a binary classification task designed to distinguish whether a given text is written by humans or generated (or revised) by a machine.

In this study, we consider detection of both *Machine-Generated Text (MGT)* and *Machine-Revised Text (MRT)*, where MRT refers to text that polished or rewritten by a model based on *Human-Written Text (HWT)*. Our study focuses on black-box detection, which better reflects real-world application than white-box settings.

Several MGTD methods have been proposed [7, 14, 19, 25, 34, 44, 49, 59] based on the assumption that the token probability distributions are distinct between MGT and HWT. Most of them leverage pre-trained language models, referred to as *scoring models*, to estimate the token probabilities of a given text and compute classification metrics for distinguishing.

Existing MGTD methods can be categorized into *zero-shot methods* [5, 7, 14, 34, 46] and *training-based methods* [9, 25].

Zero-shot methods typically rely on the inherent capabilities of scoring models [6]. However, these models are often relatively small, with limited knowledge and simple output patterns. Thus, when detecting texts deviate from their inherent distribution, such methods often struggle to achieve reliable performance.

Training-based methods use *supervised fine-tuned (SFT)* [25, 36] or *preference learning* [9, 22, 40] to align the scoring model's output distribution with that of models who built the training data.

While such approach improves detection performance for that specific models, it seems difficult to generalize this detection knowledge to models outside the training data [7, 47, 54].

We point out that both SFT and preference learning steer the scoring model toward mimicking the generators rather than being optimized directly for detection.

In other words, the training goal of previous training-based methods is model-oriented, rather than task-oriented.

This leads to the fact that the scoring model can only learn the knowledge of the generators of the training data, but cannot learn the knowledge of the detection task directly. Ultimately, it jeopardizes the generalization and robustness of the detector.

We propose **Direct Discrepancy Learning (DDL)**, a novel optimization strategy that enables the model to **learn to be a detector rather than another language model** by directly optimizing the scoring model with the output classification metric. DDL originates from the idea of freeing the scoring model from its identity as a language model and designs a task-oriented loss function so that the scoring model can directly learn the intrinsic knowledge of MGTD instead of simply fitting the distribution of training data.

Furthermore, as the fusion of prior approaches [7, 9] and DDL, we propose **DetectAnyLLM**, a unified MGTD framework. DetectAnyLLM achieves efficient and robust detection through three steps comprising *re-sampling*, *discrepancy calculation*, and *reference clustering*. Such a framework distills the core insights of existing methods [7, 34] while leveraging DDL to enhance the model's generalization capabilities and improve detection robustness.

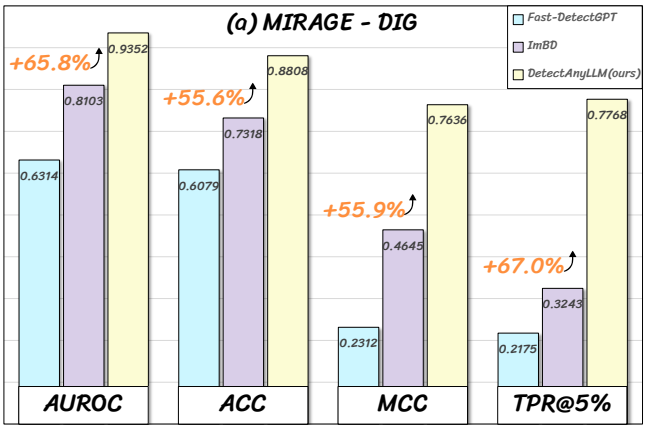

Figure 2: Performance on MIRAGE-DIG for DetectAnyLLM and State-Of-The-Art methods, including Fast-DetectGPT [7], and ImBD [9]. Imp.: $(new - old)/(1.0 - old)$**.**

Despite various MGTD studies have emerged, there remains a lack of comprehensive benchmarks [54]. Existing benchmarks [12, 21, 30, 55] suffer from several significant deficiencies: **1) Limited focus on MRT:** Most benchmark datasets, such as MGTBench [21], focus solely on MGT while neglecting the detection of MRT. **2) Narrow range of source LLMs:** Most benchmarks rely on small-scale, open-source models, whereas real-world applications often involve advanced, proprietary LLMs such as GPT-4o [24] and Claude [3]. **3) Restricted domain coverage:** Benchmarks such as HC3 [17] sample text from only one or a few domains, neglecting the domain sensitivity of machine-generated text. These deficiencies highlight a significant gap between evaluation and real-world applications. Although some recent studies [5, 54] have recognized such problems, their datasets remain insufficiently comprehensive.

To facilitate a comprehensive evaluation, we construct **MIRAGE**, the largest multi-task MGTD benchmark that provides the richest variety of proprietary LLMs and the most comprehensive text domains in MGTD research. As shown in Table 1, MIRAGE samples text from **10 corpora** in **5 common domains**, and uses **17 advanced mainstream LLMs** for text generation or revision, creating over **93K HWT-MGT pairs**. MIRAGE establishes a more realistic and reliable evaluation standard, bridging the gap between research and real-world applications.

Though existing detection methods have demonstrated seemingly outstanding performance (AUC > 0.9) on previous benchmarks [9, 55], they exhibit significant weaknesses when evaluated on MIRAGE, as Figure 2 shown. This reveals the generalization and robustness of prior methods require substantial improvement. In contrast, DetectAnyLLM still performs well, achieving an average of **0.9352** AUROC and **0.7636** MCC on the MIRAGE-DIG subset. Such performance powerfully demonstrates the efficiency and superior generalization of DDL.

Our contributions can be summarized into the following points:

- We propose **Direct Discrepancy Learning (DDL)**, a novel task-oriented optimization method that improves generalization and robustness with fewer resources and no extra data.

Table 1: Comparison between MIRAGE and existing MGTD benchmark datasets. "Size" is the capacity of the test set. "SIG" denotes Shared-Input Generation and "DIG" denotes Disjoint-Input Generation. "Commercial" refers to the use of frontier proprietary LLMs (e.g., GPT-4o). MIRAGE is the most diverse benchmark in terms of domain, tasks, and source LLMs. MIRAGE leverages powerful proprietary LLMs to generate and revise text, increasing the difficulty of detection and the realism of evaluation, enabling a more faithful evaluation of detector robustness. Furthermore, MIRAGE introduces a novel dual-scenario evaluation strategy —DIG and SIG— allowing more comprehensive assessment of both accuracy and generalization capacity.

	Data Statistic			LLMs	MGT Tasks			Other		
Benchmark	Size	Domain Coverage	Corpus	Commercial	Generate	Polish	Rewrite	Aug.	SIG	DIG
TuringBench [50]	40K	News	3	✗	✔	✗	✗	✗	✗	✔
HC3 [17]	85K	QA/Comment/Academic	5	1	✔	✗	✗	✗	-	-
M4 [53]	24.5K	QA/Comment/Academic/News	11	2	✔	✗	✗	✔	✔	✗
MAGE [30]	29K	QA/Comment/News/Academic/Story	10	3	✔	✗	✗	✔	✔	✗
RAID [11]	628.7K	News/Academic/Comment/Literature	11	3	✔	✗	✔	✔	✔	✗
DetectRL [55]	134.4K	Academic/Comment	4	2	✔	✗	✔	✔	✔	✗
HART [5]	16K	News/Literature/Academic	4	4	✔	✔	✔	✔	✗	✔
MIRAGE (ours)	93.8K	Academic/Comment/Email/News/Website	10	13	✔	✔	✔	✔	✔	✔

- We construct **MIRAGE**, a comprehensive MGTD benchmark covering diverse domains, tasks, and novelly focuses on using proprietary LLMs, resulting a more realistic evaluation.
- We present **DetectAnyLLM**, unifying prior works and DDL, achieving up to **70%** performance gains and realizing a generalizable and robust detection across domains and models.

2 Related Work

2.1 Zero-shot Detector

Previous MGTD research emphasized zero-shot detection due to concerns about overfitting during training [4, 38]. Early methods like GLTR [14] leveraged text entropy to detect machine content, while others used likelihood- or ranking-based approaches [25, 44]. Recently, DetectGPT [34] provides a novel view for MGTD, it distinguishes MGT from HWT using *perturbation*, *scoring*, and *probability curvature estimation*. Fast-DetectGPT [7] improves the perturbation step, significantly accelerating the detection process without sacrificing performance. Despite progress, zero-shot methods remain constrained by their dependence on the scoring model's output distribution, as shown by Glimpse [6], which demonstrated performance improvement through stronger scoring models.

2.2 Training-based Detector

The training-based detector fine-tunes the scoring model on specific training data. An early representative work is RoBERTa-OpenAI-Detector [44], the researchers fine-tune RoBERTa [32] models using GPT-2 [39]-generated data, performing well on detecting GPT-generated text. RADAR [23] incorporates adversarial training [15] to enhance MGTD robustness and uses PPO [42] to optimize the generator. More recently, ImBD [9] utilizes DPO [40] to optimize the scoring model based on the Fast-DetectGPT [7] framework, aiming to help the scoring model better capture the style features of the training data.

Despite these advancements, most methods simply focus on training the scoring model to approximate the source model's distribution rather than developing a dedicated MGTD detector. This introduces constraints to the scoring model during training, which are detrimental to the MGTD task.

2.3 MGTD Benchmark

Early benchmarks like Turingbench [50] focused on news articles generated by neural models, while the emergence of ChatGPT [37] shifted attention to LLM-generated text, exemplified by MGTBench[21] and HC3 [17]. Later efforts such as MAGE [30], MULTITuDE [33], and M4 [53] explored open-domain and multilingual detection. RAID [11] novelly introduced decoding strategy considerations to strengthen evaluation robustness, while DetectRL [55] examined vulnerabilities from a writing-attack perspective. However, most of these benchmarks rely on open-source models (indicating limited variousity) and focus mainly on MGT, overlooking more common real-world applications involving MRT, thus limits their applicability to real-world contexts.

HART [5] marked progress by incorporating both MGT and MRT using six advanced LLMs (only four proprietary LLMs), but it remains limited in generator diversity and domain scope. In this study, we scale up the number of generators to 17, where 13 are proprietary LLMs and 4 are advanced open-source LLMs, covering nearly all major LLMs used in real-world applications. Moreover, we sample HWT [20, 43] from five distinct domains and generate both MGT and MRT, ensuring a more comprehensive and representative evaluation. To advance MGTD research and enable fairer comparisons, we advocate for the adoption of a unified benchmark to ensure consistency in evaluation standards. We hope MIRAGE will serve as a valuable step toward achieving this goal.

3 DetectAnyLLM Framework

DetectAnyLLM builds upon Fast-DetectGPT [7], which determines whether a text is MGT by measuring the log-probability discrepancy between the original text and its perturbed variants [34]. This method involves three key steps: *1) re-sampling the given text*, *2) computing the discrepancy between original text and re-sampled text*, and *3) making a decision using the discrepancy*. DDL is utilized

to train the scoring model to enhance steps 1) and 2) so that the detector can more easily distinguish between MGT and HWT. In Section 3.1, we describe how the log-probability discrepancy is calculated. Next, in Section 3.2, we explain the motivation behind our improvements to this detection process, along with the specific designs we introduce. Finally, in Section 3.3, we detail how discrepancy is ultimately used for MGTD within our proposed framework.

3.1 Preliminary

Basic Hypothesis. Machine-generated text tends to consist of high-probability tokens at each position, whereas human-written text has greater variability. Although sampling strategies like top-k and top-p introduce some randomness, LLMs still generally select tokens with relatively high probabilities. Thus, features in the probability distribution of tokens can serve as useful cues for distinguishing machine-generated text from human-written.

Probability Discrepancy. Given a text x and a scoring model f_θ, when using a language model q_ϕ to produce perturbations, the *probability discrepancy* (i.e., probability curvature) [34] can be expressed as:

$$d(x, f_\theta, q_\phi) = \log f_\theta(x) - \mathbb{E}_{\tilde{x}\sim q_\phi(\cdot|x)}[\log f_\theta(\tilde{x})], \tag{1}$$

where $\tilde{x}$ is the perturbed version of x by q_ϕ.

Based on the hypothesis, machine-generated text x_m tends to have a high log-probability, whereas its perturbed version $\tilde{x_m}$ shows a lower log-probability. In contrast, human-written text x_h generally has a lower log-probability. When perturbed, $\tilde{x_h}$ tends to show an increase in log-probability, as the perturbation process replaces words in x_h with higher-likelihood alternatives according to the model. Thus, we expect to achieve:

$$d(x_m, f_\theta, q_\phi) > d(x_h, f_\theta, q_\phi). \tag{2}$$

This inequality forms the basis of MGTD [7, 9, 34].

When calculating this discrepancy, achieving f_θ is straightforward, allowing $\log f_\theta(x)$ to be efficiently computed. However, since the log-probabilities are computed using Markov-Chain, even a small perturbation requires recalculating the entire chain. Thus, estimating the expectation of the log-probability of $\tilde{x}$ is complex.

Conditional Probability. [7] is a biased yet computationally efficient estimation of the original probability:

$$f_\theta(\tilde{x}) = \prod_i f_\theta(\tilde{x}_i|\tilde{x}_{<i}) \sim \prod_i f_\theta(\tilde{x}_i|x_{<i}) = f_\theta(\tilde{x}|x). \tag{3}$$

By introducing Eq. (3), the probability discrepancy in Eq. (1) can be further reformulated to the *conditional probability discrepancy*:

$$d_c(x, f_\theta, q_\phi) = \frac{\log f_\theta(x|x) - \tilde{\mu}}{\tilde{\sigma}}, \tag{4}$$

where

$$\begin{aligned} \tilde{\mu} &= \mathbb{E}_{\tilde{x}\sim q_\phi(\tilde{x}|x)}[\log f_\theta(\tilde{x}|x)], \\ \tilde{\sigma}^2 &= \mathbb{E}_{\tilde{x}\sim q_\phi(\tilde{x}|x)}[\log f_\theta(\tilde{x}|x) - \tilde{\mu}^2]. \end{aligned} \tag{5}$$

Noticing that a normalization item $\tilde{\sigma}$ is added into the discrepancy function, we further explore how the $\tilde{\sigma}$ affects performance in Section 5.2.

Re-sample text. Given a sentence consisting of s tokens, we use the model q_ϕ to compute $q_\phi(t|x_{<i})$ for i from 1 to s, where t represents for token. This results in a tensor $lprobs$ of shape (s, v), where v denotes the vocabulary size of q_ϕ. With such a tensor, we can efficiently generate n re-sampled samples with only a single line of PyTorch code.

For the original version of probability discrepancy [34], both perturbation generation and discrepancy estimation require calculating a whole Markov-Chain for n times. This leads to a time complexity of $O(n \times s)$.

By introducing conditional probability [7], the resampling approach can replace the perturbation step. Under this formulation, both generating n samples and computing the discrepancy require running the Markov-Chain only **once**. As a result, the time complexity is reduced to $O(s)$.

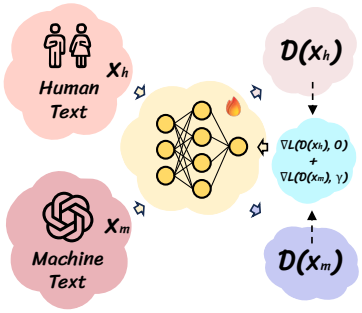

Figure 3: Overview of Direct Discrepancy Learning. The scoring model receives paired HWT-MGT data and computes the discrepancy for each. We optimize it to minimize the HWT's discrepancy while maximizing the MGT's.

3.2 Optimizing by Direct Discrepancy Learning

As shown in Eq. (4) and Eq. (2), the key to enhance the detector's performance is to increase the distribution difference of the conditional probability discrepancy between MGT and HWT estimated by the scoring model.

While ImBD [9] has achieved significant performance gains by incorporating *Direct Preference Optimization (DPO)* [40] to optimize the scoring model, we argue that DPO is not the optimal optimization method for the MGTD task.

DPO. [40] is derived from the optimization objective of *Proximal Policy Optimization (PPO)* [42], which is:

$$\max_\theta \mathbb{E}_{x\sim f_\theta(x)}[r(x)] - \beta\mathbb{D}_{KL}[f_\theta(x) \,||\, f_{ref}(x)], \tag{6}$$

where x is a text sampled from the scoring model f_θ's distribution, and r is a reward function that can judge whether this sample is bad or good. By analyzing and re-parameterizing this optimization objective, we can obtain DPO's optimization objective:

$$\max_\theta \mathbb{E}_{x_m, x_h\sim D}[\log \sigma(\beta \log \frac{f_\theta(x_m)}{f_{ref}(x_m)} - \beta \log \frac{f_\theta(x_h)}{f_{ref}(x_h)})], \tag{7}$$

where x_m denotes MGT and x_h stands for HWT. f_{ref} is a reference model, usually the original f_θ. The detailed derivation process will be presented in the supplementary material.

Motivated by redundant KL-regularization. The KL term between f_θ and f_{ref} is explicitly added to the optimization objective in PPO [42], and its weight is adjusted via β, as shown in Eq. (6).

While in DPO [40], as Eq. (7) shown, such regularization is implicitly embedded in the optimization objective, and its strength can also be adjusted by β. ImBD [9] directly adopts Eq.(7) as its loss function and leverages paired MGT-HWT data to optimize the scoring model f_θ. The KL-regularization forces the scoring model to retain its internal knowledge while learning preferences.

This leads us to question: for the MGTD task, what is the significance of retaining the original knowledge of the scoring model during training?

Since we have introduced training, our direct objective should be enable the scoring model to better capture the knowledge of the MGTD task. Fundamentally, we hope that the training process will teach the scoring model how to become a detector. However, the KL-regularization drastically shifts this objective: from learning the intrinsic knowledge of the MGTD task to aligning the scoring model with the distribution of training data. This shifts the training process from learn a detector to mimic a language model, thereby misleading the scoring model.

Direct Discrepancy Learning. Based on the reasoning above, we remove the KL-regularization in the optimization objective. Thus, the optimization goal can be re-written as:

$$\max_{\theta} \mathbb{E}_{x\sim D}[r(x)]. \tag{8}$$

We further design a simple but task-oriented reward objective $r(x)$, defined as:

$$r(x) = \begin{cases} -\|\gamma - d_c(x, f_\theta, q_\phi)\|_1, & \text{when } x \text{ is } x_m, \\ -\|d_c(x, f_\theta, q_\phi)\|_1, & \text{when } x \text{ is } x_h. \end{cases} \tag{9}$$

where γ is an hyper-parameter. This reward function is designed based on the conclusions discussed in Section 3.1, that is the discrepancy of human-written text x_h tends to be low (close to 0) while the discrepancy of machine-generated text x_m tends to be positive. The parameter γ is introduced to control how positive the discrepancy of x_m should be. In our experiment, γ is arbitrarily chosen. As shown in Table 4, an experiment on the impact of the value of γ shows that the model's performance is not particularly sensitive to this choice, indicating a level of robustness to variations in γ. In practice, our input consists of paired HWT-MGT data. We set $q_\phi = f_\theta$ following the ImBD [9]'s setting, which allows us to use the scoring model's output for optimization:

$$\min_{\theta} \mathbb{E}_{x_m, x_h\sim D}(\|d_c(x_h, f_\theta, f_\theta)\|_1 + \|\gamma - d_c(x_m, f_\theta, f_\theta)\|_1). \tag{10}$$

We call this optimization method as *Direct Discrepancy Learning (DDL)*, as it helps the scoring model directly learn the expected conditional probability discrepancy of both MGT and HWT.

By removing the KL-regularization, the scoring model can essentially forget its identity as a language model. Furthermore, the reward function based on the discrepancy d_c, which incorporates a task-oriented prior, can help the scoring model to directly learn the inherent knowledge of MGTD. Specifically, d_c for HWT approaches 0, while d_c for MGT is positive.

3.3 Detecting by Reference Clustering

We use *Reference Clustering* to achieve the transition from $d_c(x)$ to $p_m(x)$. Specifically, this algorithm is designed to estimate the probability of a given value belonging to a specific distribution, consisting of: *data aggregation* and *probability estimation*.

Data Aggregation. We first collect a certain number of MGT texts as the MGT reference dataset M, and an approximately equal number of HWT texts as the HWT reference dataset H. Then, we employ the scoring model f_θ, which will be used for detection, to respectively compute the conditional probability discrepancy d_c for each text in M and H. Thereby , we can obtain the conditional probability discrepancy distribution D_m and D_h of the texts in M and H under scored by f_θ.

Probability Estimation. We select the value in $M \cup H$ that is k_{th} closest to the target value $d_c(x)$ as the search window δ:

$$\delta = sorted(\{\|d_c(x_{ref}) - d_c(x)\|_1 \mid x_{ref} \in M \cup H\})[k], \tag{11}$$

where k is a hyper-parameter that should be determined by the size of reference dataset. For a larger reference dataset, a larger k is better as it can provide higher precision of $p_m(x)$.

Then, we count the number of MGT texts and HWT texts within the window range:

$$\begin{aligned} cnt_m &= \sum_{d\in D_m} I(d_c(x) - \delta < d < d_c(x) + \delta), \\ cnt_h &= \sum_{d\in D_h} I(d_c(x) - \delta < d < d_c(x) + \delta). \end{aligned} \tag{12}$$

Finally, we estimate the probability that text x belongs to MGT using the local statistical ratio:

$$p_m(x) = \frac{cnt_m}{cnt_m + cnt_h}. \tag{13}$$

Since the window δ is adaptively determined by the data distribution, this method can maintain stability under different data densities, thereby improving the robustness of real-world MGTD.

4 Proposed MIRAGE Benchmark

Current benchmarks exhibit notable limitations in diversity of text domains [50, 55], coverage of source LLMs [11, 53], and evaluation tasks [17, 30].

To facilitate a generalized evaluation that better reflects real-world application, we present the ***M**ulti-domain **I**nclusive **R**ealistic **A**ssessment for machine **G**enerated text d**E**tection (**MIRAGE**)* benchmark. MIRAGE constitutes the most comprehensive multi-task MGTD evaluation framework to date, incorporating both generative and revisionary text across diverse domains, employing most advanced LLMs, including 13 proprietary and 4 open-source LLMs.

4.1 Benchmark Construction

Multi-domain Sampling. Considering that LLMs exhibit varying performance across different text domains, MIRAGE samples HWT of 5 domains from 10 corpora. Detail information is presented in Supplementary Material.

Pre-Cleaning. We remove all the '\n' character to prevent the detector from identifying MGT based on the presence of the '\n' symbol. Subsequently, we filter out texts containing 100-200 words from these datasets to control for length-based detection biases.

Inclusive MGT Tasks. Following established methodologies in the [7] and [9], we designed three distinct MGT tasks: *Generate*, *Polish*, and *Rewrite*. The *Generate* task involves creating new text

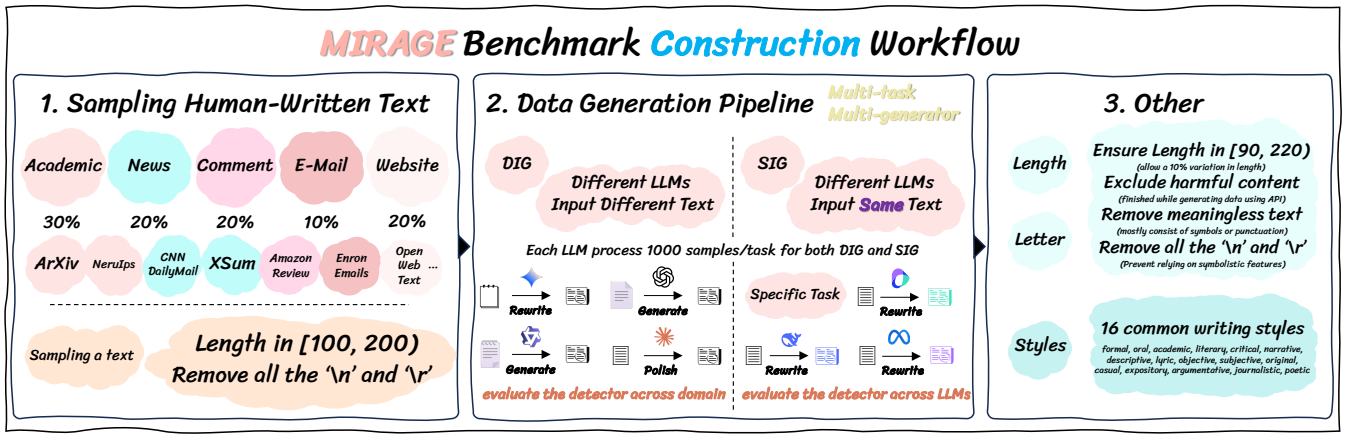

Figure 4: The MIRAGE benchmark construction workflow. MIRAGE consists of 93K HWT-MGT pairs, significantly demonstrating diversity of text-domain, source LLM, and generation task, while using writing style control as augmentation.

based on the first 30 tokens of an HWT. The *Polish* task refines an existing HWT while preserving its original details and meaning. The *Rewrite* task paraphrases a given HWT without altering its meaning or essential details. The detailed prompt of each task will be presented in Supplementary Material.

Realistic LLM Usage. In real-world applications, people typically rely on powerful proprietary LLMs to generate or revise text. However, most existing benchmarks [11, 17, 30, 50, 53, 55] rely on open-source LLMs to build data, resulting in a gap between current evaluation and real-world applications. To address this, MIRAGE incorporates 13 mainstream proprietary LLMs, as detailed in Supplementary Material.

Concurrently, recognizing the increasing deployment of high-performance open-source models in localized applications, we incorporated four advanced open-source LLMs [16, 57], ensuring comprehensive coverage of the contemporary LLM ecosystem.

Composition. We consider two distinct evaluation scenarios to better reflect real-world applications:

Disjoint-Input Generation(DIG): Each LLM generates MGT or MRT based on a unique HWT. Detectors must distinguish between this machine output and its source HWT.

Shared-Input Generation (SIG): Multiple LLMs generate MGT or MRT from the same HWT. Detectors must identify all machine outputs from a common input.

We design each LLM to generate 2,000 samples for each MGT task, equally distributed between DIG and SIG scenarios (1,000 samples each). Both DIG and SIG follow the same domain distribution for consistency, as detailed in Supplementary Material.

Sampling begins with constructing domain level HWT datasets by proportionally merging source datasets within each domain. Such dataset-mixing strategy eases dataset-bias by preventing over-sampling from single dataset.

During implementation, SIG is treated as an independent "model" and incorporates alongside the 17 individual LLMs in the sampling process. For each model (including SIG), we sequentially sample data from each domain dataset. Once sampled, items are removed from corresponding domain datasets to maintain the distinction between DIG and SIG data. Within each text domain, data is sampled continuously until the number of samples for that domain meets the requirements specified in Supplementary Material. Once the data sampling for one text domain is complete, the process moves to the next, repeating until all text domains have been sampled.

This methodology produces the DIG dataset for each LLM and a comprehensive SIG dataset, which are subsequently combined to form the complete sample set for each LLM across all tasks.

Data Augmentation. The language style is a key distinguishing feature between HWT and MGT, with a closer alignment to human language style posing a more challenging task for MGT detectors. With this consideration, we introduce data augmentation in terms of LLM's language styles. Specifically, we incorporated the phrase "in a <style> style" into the input prompt. We manually select 16 different language styles, randomly choosing one during each LLM inference to achieve style-diversity. This approach helps assess the robustness of detectors against language styles' attacks.

Post Cleaning. After generating MGT or MRT from the above HWT data, we perform data cleaning on the generated data. First, all the '\n' and '\r' are removed, to prevent detection from symbol's feature. Next, we remove texts with fewer than 90 words or more than 220 words, to prevent the impact of text length variations on detection, and finally obtain the MIRAGE benchmark dataset. The statistical results are presented in Supplementary Material.

4.2 Evaluation Metrics

Consistent with prior works [7, 9, 34], we adopt the *Area Under the Receiver Operating Characteristic Curve (AUROC)* as the primary evaluation metric. To assess the performance on specific threshold, we incorporate *TPR at a 5% false positive rate (TPR@5%)* as a supplementary metric. Furthermore, considering the MIRAGE-SIG is a class-imbalanced dataset, we additionally report the *Matthews Correlation Coefficient (MCC)* and *Balanced Accuracy* to provide a more comprehensive evaluation. Together, this diverse set of metrics provides a comprehensive assessment of detector's performance, ensuring that the evaluation reflects both theoretical completeness and real-world applicability.

Table 2: Results across three tasks (Generate, Polish, Rewrite) under two evaluation settings (MIRAGE-DIG and MIRAGE-SIG). All methods, except for RoBERTa-Base/Large, employ GPT-Neo-2.7B [8] as the scoring model. Following the experiment settings of [9], NPR [46] and DetectGPT [34] use T5-3B [41] to generate perturbations, while Fast-DetectGPT [7] utilizes GPT-J-6B [51] to generate samples. ♠ indicates a training-based method, whereas ♢ denotes a method that requires multiple model invocations. "Imp." represents Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$. Metrics: AUROC, Balanced Accuracy, MCC, and TPR@5%. DetectAnyLLM significantly outperforms all baselines across all tasks and settings.

MIRAGE-DIG (Disjoint-Input Generation)

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.4936	0.5091	0.0183	0.0147	0.4653	0.5000	0.0000	0.0214	0.4337	0.5000	0.0000	0.0148
LogRank [25]	0.4992	0.5128	0.0260	0.0220	0.4512	0.5000	0.0000	0.0195	0.4225	0.5000	0.0000	0.0132
Entropy [14]	0.6522	0.6150	0.2543	0.1099	0.5543	0.5417	0.1247	0.0954	0.5805	0.5566	0.1650	0.1189
RoBERTa-Base [32] ♠	0.5523	0.5397	0.1434	0.1250	0.4859	0.5010	0.0088	0.0460	0.5020	0.5049	0.0293	0.0569
RoBERTa-Large [32] ♠	0.4716	0.5217	0.0842	0.0871	0.5171	0.5151	0.0340	0.0633	0.5570	0.5385	0.0864	0.0895
LRR [46]	0.5215	0.5341	0.0777	0.0701	0.4081	0.5000	0.0000	0.0200	0.3930	0.5000	0.0000	0.0188
DNA-GPT [58] ♢	0.5733	0.5595	0.1196	0.0776	0.4771	0.5004	0.0110	0.0309	0.4453	0.5001	0.0080	0.0251
NPR [46] ♢	0.6120	0.6140	0.2604	0.0191	0.5071	0.5370	0.1071	0.0318	0.4710	0.5201	0.0663	0.0226
DetectGPT [34] ♢	0.6402	0.6258	0.2758	0.0275	0.5469	0.5531	0.1328	0.0355	0.5061	0.5266	0.0826	0.0283
Fast-DetectGPT [7]	0.7768	0.7234	0.4628	0.4310	0.5720	0.5570	0.1293	0.1189	0.5455	0.5432	0.1015	0.1025
ImBD [9] ♠	0.8597	0.7738	0.5497	0.4065	0.7888	0.7148	0.4300	0.2730	0.7825	0.7068	0.4139	0.2933
DetectAnyLLM (ours) ♠	**0.9525**	**0.8988**	**0.7975**	**0.7770**	**0.9297**	**0.8732**	**0.7487**	**0.7756**	**0.9234**	**0.8705**	**0.7447**	**0.7778**
Imp.	+66.14%	+55.26%	+55.03%	+62.43%	+66.71%	+55.54%	+55.91%	+69.13%	+64.78%	+55.83%	+56.44%	+68.56%

MIRAGE-SIG (Shared-Input Generation)

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.4968	0.5207	0.0196	0.0145	0.4599	0.5002	0.0030	0.0233	0.4319	0.5000	0.0000	0.0111
LogRank [25]	0.5008	0.5183	0.0182	0.0186	0.4468	0.5000	0.0000	0.0211	0.4221	0.5000	0.0000	0.0118
Entropy [14]	0.6442	0.6123	0.1592	0.1074	0.5640	0.5439	0.0516	0.0946	0.5858	0.5645	0.0918	0.1198
RoBERTa-Base [32] ♠	0.5368	0.5392	0.0529	0.1101	0.4741	0.5011	0.0048	0.0395	0.5099	0.5122	0.0221	0.0668
RoBERTa-Large [32] ♠	0.4703	0.5236	0.0417	0.0910	0.5150	0.5157	0.0283	0.0702	0.5576	0.5426	0.0405	0.0762
LRR [46]	0.5214	0.5311	0.0314	0.0657	0.4076	0.5000	0.0000	0.0238	0.3978	0.5000	0.0000	0.0174
DNA-GPT [58] ♢	0.5759	0.5647	0.0603	0.0813	0.4788	0.5001	0.0036	0.0340	0.4457	0.5002	0.0048	0.0258
NPR [46] ♢	0.6088	0.6170	0.1571	0.0185	0.5074	0.5277	0.0612	0.0293	0.4738	0.5204	0.0340	0.0177
DetectGPT [34] ♢	0.6353	0.6241	0.1719	0.0193	0.5434	0.5515	0.0668	0.0309	0.5079	0.5260	0.0431	0.0239
Fast-DetectGPT [7]	0.7706	0.7193	0.2078	0.4200	0.5727	0.5619	0.0607	0.1238	0.5480	0.5495	0.0525	0.1097
ImBD [9] ♠	0.8612	0.7791	0.5599	0.4183	0.7951	0.7199	0.4451	0.3036	0.7694	0.6920	0.3936	0.2868
DetectAnyLLM (ours) ♠	**0.9526**	**0.9059**	**0.8119**	**0.7722**	**0.9316**	**0.8740**	**0.7483**	**0.7779**	**0.9158**	**0.8643**	**0.7320**	**0.7574**
Imp.	+65.85%	+57.40%	+57.25%	+60.84%	+66.62%	+55.02%	+54.64%	+68.11%	+63.49%	+55.94%	+55.80%	+65.98%

5 Experiment

5.1 Main Results

Training settings. The scoring model and training data used in DetectAnyLLM are set exactly the same as [9] to ensure a fair comparison. Detailed training settings are provided in the Supplementary Material. The γ in DDL is set to 100, and we will discuss how the γ affects the performance in Section 5.2.

Baselines. For a comprehensive comparison, we compare the performance of our method with baseline methods, advanced zero-shot methods, and state-of-the-art training-based methods. The baseline methods including *Likelihood* [44], *Log-Rank* [25], *LRR* [46], and *Entropy* [14]. The advanced zero-shot methods includes *DetectGPT* [34], *NPR* [46], and *Fast-DetectGPT* [7]. The training-based methods includes *RoBERTa series* [32, 44] and *ImBD* [9].

Results on MIRAGE-DIG. As the top of Table 2 shown, DetectAnyLLM achieves substantial performance improvement over all baselines across all metrics and tasks. Specifically, it delivers AUROC relative gains of up to +64.78%~ +66.71%, with MCC improvements reaching up to +56.44%. DetectAnyLLM also maintains robust TPR@5% across all tasks, outperforming previous training-based SOTA ImBD [9] by large margins (+60.84%~ +69.13%).

Results on MIRAGE-SIG. As the bottom of Table 2 shown, DetectAnyLLM continues to lead in the MIRAGE-SIG subset, reaching AUROC of 0.9526, Balanced Accuracy of 0.9059, and TPR@5% up to 0.7779, again greatly surpassing all other methods.

The results on MIRAGE highlight DetectAnyLLM's strong generalization capacity and robustness across diverse source LLMs and text-domains, demonstrating the great effectiveness of DDL.

Detection on the previous test sets. We evaluate the performance of DetectAnyLLM on the three test sets used by ImBD [9]. As Table 3 shown, DetectAnyLLM consistently outperforms all existing MGTD methods.

Comparing Table 3 and Table 2, we observe that the baseline methods exhibit great performance degradation on MIRAGE. Such an observation reveals the limitations of existing test benchmarks in comprehensively evaluating detectors' abilities, underscoring the importance of MIRAGE as a more challenging benchmark.

Efficiency. Since DDL performs optimization without relying on a reference model, it achieves substantial improvements in training efficiency compared to *Style Preference Optimization (SPO)* [9]. DDL demonstrates a +30.12% reduction in training time and a +35.90%

Table 3: Results of detection on the previous test sets. ♠ indicates a training-based method, whereas ♢ denotes a method that requires multiple model invocations. "Imp." represents Improvement, computed as $(new - old)/(1.0 - old)$.

ImBD [9] Test Dataset (GPT-4o polished)			
Methods	XSum [35]	Writing [13]	PubMed [28]
Likelihood [44]	0.4396	0.8077	0.4596
LogRank [25]	0.4002	0.7694	0.4472
Entropy [14]	0.6122	0.2802	0.5899
RoBERTa-Base [32] ♠	0.4921	0.4774	0.2496
RoBERTa-Large [32] ♠	0.4782	0.4708	0.3089
LRR [46]	0.3095	0.6214	0.4710
DNA-GPT [58] ♢	0.4974	0.7478	0.3151
NPR [46] ♢	0.5065	0.8444	0.3740
DetectGPT [34] ♢	0.6217	0.8771	0.5612
Fast-DetectGPT [7]	0.6293	0.8324	0.6175
ImBD [9] ♠	0.9486	0.9468	0.7743
DetectAnyLLM (ours) ♠	**0.9880**	**0.9671**	**0.8817**
Imp.	+80.16%	+38.16%	+47.59%

reduction in memory consumption relative to SPO [9]. Details are provided in Supplementary Material.

5.2 Ablation Study

Ablation on parameter γ. As shown in Table 4, DDL exhibits strong robustness to the values of γ. Comparing to the results in Table 2, for all selected values of γ, the DDL-trained detector consistently outperforms all prior state-of-the-art methods in terms of AUROC. Detailed results, comprehensive analysis, and discussion are provided in Supplementary Material.

Table 4: Results of different γ in DDL. Metrics with subscript $_t$ correspond to the training set, and subscript $_v$ indicates evaluation on the polish task of MIRAGE-DIG.

	$\gamma = 10$	$\gamma = 20$	$\gamma = 50$	$\gamma = 100$	$\gamma = 500$	$\gamma = 10000$
AUROC_t	**0.9964**	0.9934	0.9883	0.9861	0.9861	0.9861
AUPR_t	**0.9965**	0.9938	0.9888	0.9833	0.9833	0.9833
AUROC_v	0.8692	0.9257	**0.9347**	0.9259	0.9259	0.9259
AUPR_v	0.8735	0.9294	**0.9458**	0.9373	0.9373	0.9373

Ablation on KL-strength β in SPO [9]. We provide comprehensive experiments and confirm our point in Section 3.2. For more information, please see Supplementary Material.

Ablation on model. We retrain Qwen2-0.5B [56], GPT-J-6B [51], and GPT-Neo-2.7B [8] using SPO [9] and DDL. The models are then evaluated on the *Rewrite* task of MIRAGE-SIG.

As Table 5 shown, the DDL-optimized detector consistently outperforms the SPO [9]-optimized ones across all model sizes, confirming DDL's robustness and adaptability. It is worth noting that the detector trained using a smaller but more advanced LLM, such as the Qwen2-0.5B model, achieves a better performance. This shows that the ability of the scoring model largely affects the upper limit of the detector's ability.

Ablation on normalization σ. As shown in Table 6, the removal of σ leads to a substantial degradation in performance across all metrics, highlighting the importance of σ for stable and effective optimization. Despite this, DDL without normalization still surpasses previous state-of-the-art methods on most metrics reported in Table 2, underscoring the robustness of DDL. We suggest that the normalization item σ helps standardize the output across diverse source LLMs and domains, thereby facilitating more consistent and generalizable learning.

Table 5: Ablation study on the impact of scoring model. All models are trained on the same data and evaluated on the MIRAGE-SIG *Rewrite* task. Improvement is marked red.

Method	Base Model	AUROC	Accuracy	MCC	TPR@5%
SPO [9]	Qwen2-0.5B [56]	0.8570	0.7816	0.5632	0.4508
	GPT-Neo-2.7B [8]	0.7694	0.6920	0.3936	0.2868
	GPT-J-6B [51]	0.8367	0.7557	0.5155	0.4722
DDL (ours)	Qwen2-0.5B [56]	0.9370 +55.94%	0.9071 +57.46%	0.8169 +58.08%	0.8575 +74.05%
	GPT-Neo-2.7B [8]	0.9158 +63.49%	0.8643 +55.94%	0.7320 +55.80%	0.7574 +65.98%
	GPT-J-6B [51]	0.8909 +33.19%	0.8424 +35.49%	0.6878 +35.56%	0.6047 +25.10%

Table 6: Ablated result of σ. "norm." means "normalization". "w/" means "with" and "w/o" means "without". Scoring model: GPT-Neo-2.7B [8]. Benchmark: MIRAGE-DIG-polish.

	AUROC	Accuracy	MCC	TPR@5%
$\text{DDL}_{\text{w/o } norm.}$	0.8499	0.7759	0.5563	0.5232
$\text{DDL}_{\text{w/ } norm.}$	**0.9297**	**0.8732**	**0.7487**	**0.7756**

6 Conclusion

In this study, we have introduced a novel optimization strategy, *Direct Discrepancy Learning (DDL)*, and developed a unified detection framework named *DetectAnyLLM*. Our approach enables the scoring model to acquire task-oriented knowledge by directly leveraging discrepancy signals and achieves high-precision detection through a technique we call *reference clustering*. We also proposed *MIRAGE*, a comprehensive benchmark dataset that spans a wide range of text domains, the most advanced LLMs, and generation tasks. To thoroughly evaluate detector performance, we assessed DetectAnyLLM and existing MGTD methods under two settings: *Disjoint-Input Generation* and *Shared-Input Generation*. Experimental results on both MIRAGE and previously established test sets demonstrate that DetectAnyLLM significantly outperforms existing MGTD methods, establishing a new state-of-the-art in this domain.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (62306153, 62225604), the Natural Science Foundation of Tianjin, China (24JCJQJC00020), the Young Elite Scientists Sponsorship Program by CAST (YESS20240686), the Fundamental Research Funds for the Central Universities (Nankai University, 070-63243143), and Shenzhen Science and Technology Program (JCYJ20240813114237048).

The computational devices are partly supported by the Supercomputing Center of Nankai University (NKSC).

References

[1] Mervat Abassy, Kareem Elozeiri, Alexander Aziz, Minh Ta, Raj Tomar, Bimarsha Adhikari, Saad Ahmed, Yuxia Wang, Osama Mohammed Afzal, Zhuohan Xie, et al. 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*. 336–343.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774* (2023).

[3] Anthropic. 2024. *Model Card Addendum: Claude 3.5 Haiku and Upgraded Claude 3.5 Sonnet*. https://assets.anthropic.com/m/1cd9d098ac3e6467/original/Claude-3-Model-Card-October-Addendum.pdf

[4] Anton Bakhtin, Sam Gross, Myle Ott, Yuntian Deng, Marc'Aurelio Ranzato, and Arthur Szlam. 2019. Real or fake? learning to discriminate machine from human generated text. *arXiv preprint arXiv:1906.03351* (2019).

[5] Guangsheng Bao, Lihua Rong, Yanbin Zhao, Qiji Zhou, and Yue Zhang. 2025. Decoupling Content and Expression: Two-Dimensional Detection of AI-Generated Text. arXiv:cs.CL/2503.00258 https://arxiv.org/abs/2503.00258

[6] Guangsheng Bao, Yanbin Zhao, Juncai He, and Yue Zhang. 2025. Glimpse: Enabling White-Box Methods to Use Proprietary Models for Zero-Shot LLM-Generated Text Detection. In *The Thirteenth International Conference on Learning Representations, ICLR*.

[7] 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 *ICLR*.

[8] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. 2021. *GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow*. https://doi.org/10.5281/zenodo.5297715

[9] Jiaqi Chen, Xiaoye Zhu, Tianyang Liu, Ying Chen, Chen Xinhui, Yiwen Yuan, Chak Tou Leong, Zuchao Li, Long Tang, Lei Zhang, et al. 2025. Imitate Before Detect: Aligning Machine Stylistic Preference for Machine-Revised Text Detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 39. 23559–23567.

[10] Yao Dou, Maxwell Forbes, Rik Koncel-Kedziorski, Noah A Smith, and Yejin Choi. 2021. Is GPT-3 text indistinguishable from human text? scarecrow: A framework for scrutinizing machine text. *arXiv preprint arXiv:2107.01294* (2021).

[11] 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)*. 12463–12492.

[12] Liam Dugan, Daphne Ippolito, Arun Kirubarajan, Sherry Shi, and Chris Callison-Burch. 2023. Real or fake text?: Investigating human ability to detect boundaries between human-written and machine-generated text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 12763–12771.

[13] Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical Neural Story Generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Iryna Gurevych and Yusuke Miyao (Eds.). Association for Computational Linguistics, Melbourne, Australia, 889–898. https://doi.org/10.18653/v1/P18-1082

[14] Sebastian Gehrmann, Hendrik Strobelt, and Alexander M 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*. 111–116.

[15] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. *Advances in neural information processing systems* 27 (2014).

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

[17] 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* (2023).

[18] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948* (2025).

[19] 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*. 17519–17537.

[20] Tahmid Hasan, Abhik Bhattacharjee, Md Saiful Islam, Kazi Mubasshir, Yuan-Fang Li, Yong-Bin Kang, M Sohel Rahman, and Rifat Shahriyar. 2021. XL-Sum: Large-Scale Multilingual Abstractive Summarization for 44 Languages. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. 4693–4703.

[21] Xinlei He, Xinyue Shen, Zeyuan Chen, Michael Backes, and Yang Zhang. 2024. Mgtbench: Benchmarking machine-generated text detection. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 2251–2265.

[22] Jiwoo Hong, Noah Lee, and James Thorne. 2024. ORPO: Monolithic Preference Optimization without Reference Model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. 11170–11189.

[23] Xiaomeng Hu, Pin-Yu Chen, and Tsung-Yi Ho. 2023. Radar: Robust ai-text detection via adversarial learning. *Advances in neural information processing systems* 36 (2023), 15077–15095.

[24] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276* (2024).

[25] Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2019. Automatic detection of generated text is easiest when humans are fooled. *arXiv preprint arXiv:1911.00650* (2019).

[26] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720* (2024).

[27] Ganesh Jawahar, Muhammad Abdul-Mageed, and VS Laks Lakshmanan. 2020. Automatic Detection of Machine Generated Text: A Critical Survey. In *Proceedings of the 28th International Conference on Computational Linguistics*. 2296–2309.

[28] Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A Dataset for Biomedical Research Question Answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, Hong Kong, China, 2567–2577. https://doi.org/10.18653/v1/D19-1259

[29] 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*. 17036–17055.

[30] 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)*. 36–53.

[31] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437* (2024).

[32] 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* (2019).

[33] Dominik Macko, Robert Moro, Adaku Uchendu, Jason Lucas, Michiharu Yamashita, Matúš Pikuliak, Ivan Srba, Thai Le, Dongwon Lee, Jakub Simko, et al. 2023. MULTITuDE: Large-Scale Multilingual Machine-Generated Text Detection Benchmark. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. 9960–9987.

[34] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. In *International Conference on Machine Learning*. PMLR, 24950–24962.

[35] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii (Eds.). Association for Computational Linguistics, Brussels, Belgium, 1797–1807. https://doi.org/10.18653/v1/D18-1206

[36] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[37] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems* 35 (2022), 27730–27744.

[38] Jiameng Pu, Zain Sarwar, Sifat Muhammad Abdullah, Abdullah Rehman, Yoonjin Kim, Parantapa Bhattacharya, Mobin Javed, and Bimal Viswanath. 2023. Deepfake text detection: Limitations and opportunities. In *2023 IEEE symposium on security and privacy (SP)*. IEEE, 1613–1630.

[39] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. *Language models are unsupervised multitask learners*.

[40] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing*

Systems 36 (2023), 53728–53741.

[41] 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 (2020), 1–67.

[42] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017).

[43] Eva Sharma, Chen Li, and Lu Wang. 2019. BIGPATENT: A Large-Scale Dataset for Abstractive and Coherent Summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 2204–2213.

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

[45] Mayank Soni and Vincent Wade. 2023. Comparing abstractive summaries generated by ChatGPT to real summaries through blinded reviewers and text classification algorithms. *arXiv preprint arXiv:2303.17650* (2023).

[46] 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*. 12395–12412.

[47] Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. 2024. The science of detecting LLM-generated text. *Commun. ACM* 67, 4 (2024), 50–59.

[48] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805* (2023).

[49] 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. In *Proceedings of the 37th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) *(NIPS '23)*. Curran Associates Inc., Red Hook, NY, USA, Article 1706, 20 pages.

[50] Adaku Uchendu, Zeyu Ma, Thai Le, Rui Zhang, and Dongwon Lee. 2021. TURINGBENCH: A Benchmark Environment for Turing Test in the Age of Neural Text Generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*. 2001–2016.

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

[52] William Yang Wang. 2017. " Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics.

[53] Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, et al. 2024. M4: Multi-generator, Multi-domain, and Multi-lingual 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)*. 1369–1407.

[54] Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. 2025. A survey on LLM-generated text detection: Necessity, methods, and future directions. *Computational Linguistics* (2025), 1–66.

[55] Junchao Wu, Runzhe Zhan, Derek Wong, Shu Yang, Xinyi Yang, Yulin Yuan, and Lidia Chao. 2024. Detectrl: Benchmarking llm-generated text detection in real-world scenarios. *Advances in Neural Information Processing Systems* 37 (2024), 100369–100401.

[56] An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. 2024. Qwen2 Technical Report. *CoRR* (2024).

[57] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115* (2024).

[58] Xianjun Yang, Wei Cheng, Yue Wu, Linda Ruth Petzold, William Yang Wang, and Haifeng Chen. 2024. DNA-GPT: Divergent N-Gram Analysis for Training-Free Detection of GPT-Generated Text. In *ICLR*.

[59] Xiao Yu, Kejiang Chen, Qi Yang, Weiming Zhang, and Nenghai Yu. 2024. Text fluoroscopy: Detecting LLM-generated text through intrinsic features. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. 15838–15846.

Table 7: Source datasets for each domain.

Domains	Datasets
Academic	BigPatent [43], NeurIPS, ArXiv, PubMed-Abstracts [28]
eMail	Enron-Emails
Website	OpenWebText
News	CNNDailyMails, XSum [35], XLSum [20]
Comment	Amazon-Review

A More details on DDL

A.1 Derivation of DPO

Direct Preference Learning (DPO) [40] is derived from the optimization objective of *Proximal Policy Optimization (PPO)* [42], which is formulated as:

$$\max_{\theta} \mathbb{E}_{x\sim f_\theta(x)}[r(x)] - \beta\mathbb{D}_{KL}[(f_\theta(x) \,||\, f_{ref}(x)], \tag{14}$$

where x is a text sampled from the scoring model f_θ's distribution, and r is a reward function that can judge whether this sample is bad or good. We can obtain DPO's optimization objective by analyzing and re-parameterizing this optimization objective. The f_{ref} stands for the reference model, usually the original f_θ. DPO [40] starts with explicit solution $f_\theta = p_r$ of Eq. (14):

$$p_r(x) = \frac{1}{Z(x)} f_{ref}(x) \exp(\frac{1}{\beta} r(x)), \tag{15}$$

where:

$$Z(x) = \sum_x f_{ref}(x) \exp(\frac{1}{\beta} r(x)). \tag{16}$$

Furthermore, we can reparameterize r as:

$$r(x) = \beta \log \frac{p_r(x)}{f_{ref}(x)} + \beta \log Z(x), \tag{17}$$

where, p_r is the best solution of Eq. (14), which we want f_θ to become. Now, if we introduce the Bradley-Terry model to present the model's preference L between HWT x_h and MGT x_m, we can get:

$$\begin{aligned} L(x_m \succ x_h) &= \sigma(r(x_m) - r(x_h)) \\ &= \sigma(\beta \log \frac{p_r(x_m)}{f_{ref}(x_m)} - \beta \log \frac{p_r(x_h)}{f_{ref}(x_h)}), \end{aligned} \tag{18}$$

where we surprisingly reduced the partition function $Z(x)$. By replacing p_r with f_θ and utilizing Maximum-Likelihood Estimation to Eq. (18), we can finally present the DPO [40] optimize goal :

$$\max_{\theta} \mathbb{E}_{x_m, x_h\sim D}[\log \sigma(\beta \log \frac{f_\theta(x_m)}{f_{ref}(x_m)} - \beta \log \frac{f_\theta(x_h)}{f_{ref}(x_h)})], \tag{19}$$

where x_m denotes MGT and x_h stands for HWT. f_{ref} is a reference model, usually the original f_θ.

B More details on MIRAGE

B.1 More details on Data Source

Time Bound. To ensure that all sampled texts are human-written and free from contamination by LLM-generated content, most of the source datasets used were constructed prior to 2021. For datasets containing data collected after 2021, we cleaned the data denoted collected after 2021 in these datasets to ensure the purity and authenticity of the human-written source material.

Source Domains and Datasets. MIRAGE encompasses a diverse range of text domains, including *Academic*, *E-Mail*, *Website*, *News*, and *Comment*. The mapping between these domains and the corresponding source datasets is summarized in Table 7.

Additionally, we implement domain-specific pre-processing: extracting only abstracts from academic publications (NeurIPS and ArXiv) and isolating message content from email communications (Enron-Emails dataset).

Domain Composition. As the amount of data varies across domains, the text domains are not treated equally in terms of quantity. However, for both DIG and SIG, the proportion of texts from each domain that each LLM is required to generate or revise remains fixed. The detailed domain distribution is shown in Table 8.

Table 8: Text domain composition that each LLM is required to perform generation task in both DIG and SIG.

Academic	Mail	Website	News	Comment	**Total**
300	100	200	200	200	1000

Statistic Result. Table 9 presents the overall statistics of the MIRAGE dataset across the two task settings: Disjoint-Input Generation and Shared-Input Generation. For each setting, the dataset includes three task types—*Generate* (Gen.), *Polish* (Pol.), and *Rewrite* (Rew.). The number of instances is balanced across both settings, with each task type containing approximately 14,000 to 16,000 samples. This balanced distribution ensures that the dataset supports a comprehensive evaluation of LLMs across different generation and revision scenarios.

Table 9: Statistic result of MIRAGE.

	Disjoint-Input Generation			Shared-Input Generation		
Tasks	Gen.	Pol.	Rew.	Gen.	Pol.	Rew.
Count	16412	14776	15735	16388	14776	15751

B.2 Source Generator LLMs

The source LLMs used for data generation in MIRAGE are listed in Table 10. In total, MIRAGE samples machine-generated texts (MGT) using 13 powerful commercial LLMs and 4 advanced open-source LLMs. This selection reflects a strong emphasis on evaluating detection performance in real-world applicant scenarios, while still maintaining attention to the open-source LLM landscape.

Table 10: Commercial LLMs are highlighted in bold.

Series	Models
GPT	**GPT-4o** [24], **GPT-o3-mini** [26], **GPT-4o-mini** [24]
Claude	**Claude-3.5-Haiku**, **Claude-3.7-sonnet** [3]
DeepSeek	**DeepSeek-V3** [31], **DeepSeek-R1** [18]
Gemini	**Gemini-2.0-Flash**, **Gemini-2.0-Flash-Lite** [48]
Qwen	Qwen-2.5-7B [57], Qwen-2.5-7B-R1-Distill [18], **QwQ-Plus**
LlaMa	LlaMA-3.1-8B [16], LlaMa-3.1-8B-R1-Distill [18]
Grok	**Grok2**
Moonshot	**Moonshot-v1**
Doubao	**Doubao-1.5-pro-32k**

Table 11: Detail style list.

Style List
formal, oral, academic, literary, critical, narrative, descriptive, lyric, objective, subjective, original, casual, expository, argumentative, journalistic, poetic

B.3 Prompts for Generation Tasks

The system prompts used for all three generation tasks are the same, specifically, "You are a professional writing assistant who can write high-quality, coherent, and engaging articles. " We add a style control signal to the user prompt, in order to perform data augmentation, thus promoting the robustness of our benchmark.

Style Control. The style control signal is directly added to the user prompt, as "in a <style> style". The <style> is randomly chosen from a prepared style list, detail as shown in Table 11

Prompt for Generate. "Write an article about 150 words in a <style> style starting exactly with: <original>". The <original> is the first 30 tokens of a HWT.

Prompt for Polish. "Polish the following text in a style style without missing any original details. Ensure that the length of the polished text is similar to the original text. Directly output your polished text. Here is the original text: original" The <original> is a complete HWT.

Prompt for Rewrite. "Paraphrase the following text in a style style without missing any original details. Ensure that the length of the paraphrased text is similar to the original text. Directly output your paraphrased text. Here is the original text: original" The <original> is a complete HWT.

B.4 More Details on Evaluation Metrics

Consistent with prior work [7, 9], we adopt *the Area Under the Receiver Operating Characteristic Curve (AUROC)* as the primary evaluation metric for assessing the performance of the MGT Detector. While AUROC provides a threshold-independent measure of classification capability, it does not necessarily reflect the detector's effectiveness at specific operating points, which are often critical in real-world deployments.

To address this limitation, we incorporate *TPR at a 5% false positive rate (TPR@5%)* as an important supplementary metric. TPR@5% reflects the detector's sensitivity when operating under a strict false positive constraint, which is especially important for applications demanding high precision.

Furthermore, considering the MIRAGE-SIG is a class-imbalanced dataset, we additionally report the *Matthews Correlation Coefficient (MCC)* and *Balanced Accuracy* to provide a more comprehensive evaluation. MCC captures the overall quality of binary classifications by considering all four elements of the confusion matrix, making it particularly informative under class imbalance. Balanced Accuracy, used in place of standard accuracy, is computed as the arithmetic mean of the true positive rate and true negative rate, making it better suited for evaluating performance on imbalanced datasets.

Together, this diverse set of metrics provides a comprehensive assessment of the detector's performance, ensuring that the evaluation reflects both theoretical completeness and real-world applicability.

C More details on Experiment

C.1 Experiment Setup

Device. All of our experiments are conducted in Linux 4.18.0(CentOS 7), using a single NVIDIA A40 GPU with 48GB GPU memory. The Python version is 3.10.16, the PyTorch version is 2.5.1, the Transformers version is 4.47.1, and the Datasets version is 3.2.0.

Training Dataset. We train DetectAnyLLM in the dataset used in ImBD [9], specifically, 500 pairs of HWT-MGT data, where MGT is machine-polished text created by GPT-3.5-Turbo.

LoRA Config. Following the settings of ImBD [9], we adopt a LoRA configuration specifically designed for causal language modeling, with a rank of 8, a LoRA alpha of 32, and a dropout rate of 0.1.

Settings for Reproducing ImBD. We reproduced ImBD [9] for comparative evaluation, following the original training configuration described in the paper. Specifically, we set the learning rate to 0.0001 and used a beta coefficient of 0.05. The only modification made was increasing the number of training epochs from 2 (as reported in the original paper) to 5, in order to ensure full convergence of the model. Throughout the training process, we monitored the model's performance on the validation set to prevent overfitting and to verify that the reproduced ImBD model maintained comparable performance to the original.

Settings for Training DetectAnyLLM. We train DetectAnyLLM using the exact same hyperparameters as those used in our reproduction of ImBD [9], including a learning rate of 0.0001 and 5 training epochs. For the optimization objective in *Direct Discrepancy Learning (DDL)*, we set the hyperparameter γ to 100. That is because increasing γ beyond this value did not lead to further improvements in performance, suggesting that the model had fully converged. Moreover, since the model's performance remains stable for larger values of γ, this setting also ensures compatibility with varying training environments, as the optimal value of γ is unknown, we simply choose a sufficiently large value that provides a safe and generalizable configuration.

C.2 Empirical Validation of Redundant KL-Regularization

To evaluate the impact of the KL-regularization term in DPO-style training, we conduct ablation studies on a wide range of β values, which directly control the strength of the implicit KL constraint.

Table 12: Detail Results of different β in SPO [9]. Metrics with subscript $_t$ correspond to the training set, and subscript $_v$ indicates evaluation on the polish task of MIRAGE-DIG. Avg.D(*) denotes the average discrepancy of *, where $\mathbf{x}_h$ stands for human-written text and $\mathbf{x}_m$ stands for machine-generated text. ΔD denotes the distance between Avg.D($\mathbf{x}_h$) and Avg.D($\mathbf{x}_m$), a higher ΔD is commonly better for discriminating between $\mathbf{x}_h$ and $\mathbf{x}_m$.

β	0.05	0.10	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50	0.60	0.70	0.80	0.90	0.95
AUROC$_t$	**0.9490**	0.9192	0.9088	0.9009	0.8945	0.8920	0.8888	0.8871	0.8821	0.8786	0.8742	0.8700	0.8622	0.8554	0.8542
AUPR$_t$	**0.9566**	0.9277	0.9155	0.9058	0.8982	0.8966	0.8935	0.8934	0.8897	0.8869	0.8846	0.8805	0.8741	0.8695	0.8689
Avg.D(x_h)$_t$	-29.14	-13.50	-8.68	-5.72	-4.47	-3.57	-2.92	-2.36	-2.06	-2.05	-1.54	-1.51	-1.37	-0.98	-0.93
Avg.D(x_m)$_t$	-11.55	-6.56	-3.88	-2.11	-1.37	-0.79	-0.39	-0.01	0.17	0.14	0.50	0.50	0.53	0.83	0.86
ΔD	17.59	6.94	4.8	3.61	3.10	2.78	2.53	2.35	2.23	2.19	2.04	2.01	1.90	1.81	1.79
AUROC$_v$	**0.7888**	0.7273	0.7045	0.6669	0.6801	0.6786	0.6745	0.6625	0.6688	0.6672	0.6520	0.6644	0.6499	0.6315	0.6477
AUPR$_v$	**0.7756**	0.7122	0.6910	0.6602	0.6783	0.6757	0.6764	0.6663	0.6778	0.6719	0.6597	0.6723	0.6578	0.6365	0.6566
Avg.D(x_h)$_v$	-26.28	-17.65	-11.50	-7.30	-5.30	-3.78	-2.67	-1.82	-1.33	-1.05	-0.47	-0.45	-0.26	0.32	0.34
Avg.D(x_m)$_v$	-18.91	-14.53	-9.94	-6.27	-4.51	-3.11	-2.06	-1.20	-0.74	-0.51	-0.00	0.00	0.10	0.78	0.82
ΔD	7.37	3.12	1.56	1.03	0.79	0.67	0.61	0.62	0.59	0.54	0.47	0.45	0.36	0.46	0.48

Table 13: Training time cost comparison between DDL and SPO [9]. The results are tested in ImBD [9]'s training dataset. Device: single NVIDIA A40. Model: GPT-J-6B [51]. "Imp." represents Improvement, computed as -(new - old) / (old).

Optim.	Batch Size	Time Cost/Epoch	Memory Usage
SPO [9]	1	166s	31.45GB
DDL(ours)	1	**116s**	**20.16GB**
Imp.	–	**+30.12%**	**+35.90%**

According to the formulation in Section 3.2, a larger β enforces stronger alignment between the scoring model f_θ and the reference model f_{ref}, effectively constraining the learning objective toward distributional conformity rather than task-specific discriminability.

Table 12 provides compelling empirical evidence supporting our hypothesis that KL-regularization can be redundant—or even detrimental—for the MGTD task. As β increases, we observe a consistent and significant degradation in detection performance across all evaluation metrics. For instance, the AUROC and AUPR on the training set drop from **0.9490**/**0.9566** at $\beta = 0.05$ to 0.8542/0.8689 at $\beta = 0.95$. Similar trends are observed on the validation set, where AUROC decreases from 0.7888 to 0.6477, and AUPR from 0.7756 to 0.6566.

At low β values, the discrepancy between D(x_h) and D(x_m) is substantial (e.g., 17.59 on the training set at $\beta = 0.05$), which allows the scoring model to effectively differentiate between natural and perturbed sequences. As β increases, this margin rapidly collapses—dropping below 3.0 by $\beta = 0.30$, and approaching near-zero at higher values. The validation set exhibits a parallel pattern: the discrepancy narrows from 7.37 at $\beta = 0.05$ to merely 0.48 at $\beta = 0.95$.

These results indicate that strong KL-regularization impairs the model's ability to learn task-oriented discriminative signals from training data. Instead of becoming a better detector, the scoring model is constrained to behave like a generic language model, limiting its effectiveness in distinguishing machine-generated text from human-written. This validates our theoretical intuition: while the KL term may serve to preserve internal knowledge in generic preference modeling tasks, it is counterproductive in MGTD.

C.3 More Details on Main Results

Efficiency Improvement. When using the scoring model f_θ as the sampling model q_ϕ, as discussed in Section 3.2, DDL eliminates the need to load a separate reference model during training, unlike SPO [9]. This design enables DDL to train with a single model, leading to notable improvements in training efficiency. Table 13 presents a detailed comparison of training time and memory usage between SPO [9] and DDL.

SPO [9] requires loading two large models simultaneously during training, resulting in high memory demands—specifically, 31.45GB for training with GPT-J-6B [51]. This exceeds the capacity of many commonly available GPUs. In contrast, DDL only requires 20.16GB of memory, making it feasible to train on widely accessible GPUs.

C.4 Discussion of γ in DDL

Detail Look of γ's effect. Table 14 shown, although there are clear performance peaks at specific values (e.g., $\gamma = 5$ for the training set and $\gamma = 30$–40 for the validation set), the metrics remain consistently high across a wide range of γ values. For instance, even when γ increases from 10 to 10000, AUROC$_t$ and AUPR$_t$ only experience a minor drop (from around 0.9964 to 0.9861), and AUROC$_v$ and AUPR$_v$ stay stable after reaching their respective optima.

Meanwhile, AUROC$_v$ and AUPR$_v$ continue to improve up to $\gamma = 30$–$\gamma = 40$, reaching their peaks at $\gamma = 30$ (AUPR$_v$ = 0.9472) and $\gamma = 40$ (AUROC$_v$ = 0.9377), after which they plateau.

DDL's Robustness on γ. While the average discrepancies and training metrics vary as γ increases, the evaluation performance (AUROC$_v$ and AUPR$_v$) remains relatively stable across a broad range—from $\gamma = 30$ to $\gamma = 10000$. For example, AUROC$_v$ fluctuates within a narrow band around 0.93, and AUPR$_v$ stays above 0.93 even when γ changes by orders of magnitude. This indicates that although γ influences how positive the discrepancy of x_m should be, the model's downstream generalization ability is not overly sensitive to its exact value.

This plateauing effect indicates that once γ surpasses a moderate threshold, the method maintains strong performance without being overly sensitive to further increases. Moreover, the saturation of Avg.D(x_h) and Avg.D(x_m) after a certain point suggests the model's behavior becomes consistent, avoiding unstable shifts.

Table 14: Detail Results of different γ in DDL. Metrics with subscript $_t$ correspond to the training set, and subscript $_v$ indicates evaluation on the polish task of MIRAGE-DIG. Avg.D(*) denotes the average discrepancy of *, where $\mathbf{x}_h$ stands for human-written text and $\mathbf{x}_m$ stands for machine-generated text.

	$\gamma = 1$	$\gamma = 2$	$\gamma = 5$	$\gamma = 10$	$\gamma = 20$	$\gamma = 30$	$\gamma = 40$	$\gamma = 50$	$\gamma = 60$	$\gamma = 70$	$\gamma = 80$	$\gamma = 90$	$\gamma = 100$	$\gamma = 500$	$\gamma = 10000$
AUROC_t	0.9501	0.9910	**0.9983**	0.9964	0.9934	0.9900	0.9886	0.9883	0.9880	0.9879	0.9861	0.9861	0.9861	0.9861	0.9861
AUPR_t	0.9379	0.9910	**0.9983**	0.9965	0.9938	0.9911	0.9865	0.9888	0.9852	0.9852	0.9833	0.9833	0.9833	0.9833	0.9833
$\text{Avg.D}(\text{x}_h)_t$	0.07	0.14	0.20	0.27	0.54	0.80	1.08	1.45	1.55	1.73	1.91	1.91	1.91	1.91	1.91
$\text{Avg.D}(\text{x}_m)_t$	0.95	1.88	4.86	8.92	17.24	24.64	31.82	37.93	40.81	41.92	42.12	42.12	42.12	42.12	42.12
AUROC_v	0.5481	0.6000	0.7833	0.8692	0.9257	0.9360	**0.9377**	0.9347	0.9251	0.9270	0.9259	0.9259	0.9259	0.9259	0.9259
AUPR_v	0.5206	0.5562	0.7452	0.8735	0.9294	**0.9472**	0.9461	0.9458	0.9401	0.9382	0.9373	0.9373	0.9373	0.9373	0.9373
$\text{Avg.D}(\text{x}_h)_v$	1.1	1.24	1.76	0.84	3.11	3.37	4.66	4.62	4.72	5.36	5.23	5.23	5.23	5.23	5.23
$\text{Avg.D}(\text{x}_m)_v$	1.36	1.85	4.86	7.11	16.09	24.22	32.96	34.86	36.75	39.86	39.43	39.43	39.43	39.43	39.43

This is a desirable property in practical scenarios, where tuning hyperparameters like γ might be challenging or resource-intensive.

Explanation of γ's Effectiveness. The robustness of our method to the hyperparameter γ arises naturally from the design of the *Direct Discrepancy Learning (DDL)*. In DDL, γ serves as a margin that guides the optimization: it encourages the model to keep the discrepancy score $\text{D}(\text{x}_m)$ of MGT close to γ, while minimizing the discrepancy score $\text{D}(\text{x}_h)$ for HWT toward zero. This setup constructs an explicit separation objective in the discrepancy space between HWT and MGT.

*We realized the goal of **learning to be a detector rather than another language model** by introducing such explicit separation objective.*

A small γ (e.g., 1–5) still may encourage separation, however, may not provide enough margin, leading to overlap between HWT and MGT discrepancies. As γ increases, the model is encouraged to push $\text{D}(\text{x}_m)$ further away from zero, thus improving distinguishability and enhancing performance—especially noticeable in the rise of AUROC/AUPR metrics from $\gamma = 1$ to $\gamma = 5$ and beyond.

Explanation of Performance Plateau. Once γ exceeds a certain threshold (e.g., $\gamma \geq 30$), we observe that performance metrics (both AUROC_t and AUROC_v) are saturate. This indicates that the discrepancy between HWT and MGT has reached a sufficient margin: $\text{D}(\text{x}_h)$ is consistently near 0 and $\text{D}(\text{x}_m)$ is already large enough. Increasing γ further only increases the target discrepancy for x_m without changing the classification boundary. The model thus stabilizes, as it can no longer extract additional useful separation from a larger γ. This explains the observed robustness—DDL achieves effective separation and remains performance over a wide range of γ values.

Reason of Setting γ to 100. We choose $\gamma = 100$ in our main results for two key reasons. First, as shown in the ablation results, performance has already plateaued by $\gamma = 100$, meaning this setting offers high performance while avoiding sensitivity to further hyperparameter tuning—making it a practical and reliable choice in real-world applications. Second, a higher γ ensures a clear and consistent discrepancy margin, improving interpretability and stability across diverse datasets or models. In deployment scenarios, where extensive hyperparameter sweeps may be infeasible, such robustness is critical.

Summary. As shown in Table 14, there exists a threshold value t_h such that the performance remains stable for all $\gamma \geq t_h$. In contrast, setting γ too small can lead to significantly degraded performance, while increasing γ beyond t_h does not cause any sharp decline. Therefore, we recommend setting γ to a relatively large value, since in real-world applications the optimal value is typically unknown.

C.5 Detection Results on specific LLM

We expand the main results in Section 5.1 in the specific LLM level to obtain the specific detection capabilities of different methods on texts generated by specific LLMs.

Results. As shown in the following tables, DetectAnyLLM achieves consistently strong performance across all metrics, domains, tasks, and source LLMs. On the Polish and Rewrite tasks, it outperforms previous state-of-the-art methods by an average margin of nearly 70%. In certain settings involving text generated by specific LLMs, FastDetectGPT[7] and ImBD[9] slightly outperform DetectAnyLLM. We attribute this to the relative simplicity of the Generate task, which allows earlier methods to perform competitively. Even in these cases, DetectAnyLLM trails by no more than 10^{-2} AUROC, suggesting that may have reached saturation on this task.

As shown in Table 15 and Table 18, the AUROC on DIG text polished by Claude-3.5-Haiku reaches **0.9903**, while that of Claude-3.7-Sonnet is **0.9096**. Similarly, the AUROC on SIG text rewritten by GPT-4o-mini reaches **0.9176**, compared to **0.8697** for GPT-4o. These results indicate that detecting text from smaller LLMs is generally easier than from their larger counterparts.

Table 15: Generator: GPT-4o, GPT-4o-mini. "Imp.": Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$.

MIRAGE-DIG, GPT-4o

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.5021	0.5476	0.1117	0.0040	0.4732	0.5052	0.0227	0.0216	0.4281	0.5000	0.0000	0.0134
LogRank [25]	0.5053	0.5406	0.1011	0.0050	0.4638	0.5005	0.0227	0.0216	0.4191	0.5000	0.0000	0.0113
Entropy [14]	0.6414	0.6259	0.2976	0.1103	0.5358	0.5402	0.1002	0.0670	0.5733	0.5520	0.1953	0.1236
RoBERTa-Base [32] ♠	0.4950	0.5206	0.0857	0.0843	0.5013	0.5036	0.0259	0.0474	0.5244	0.5335	0.0767	0.0639
RoBERTa-Large [32] ♠	0.4478	0.5005	0.0224	0.0361	0.5715	0.5603	0.1207	0.0732	0.6244	0.5896	0.1849	0.0865
LRR [46]	0.5172	0.5281	0.0562	0.0130	0.4334	0.5000	0.0000	0.0206	0.3957	0.5000	0.0000	0.0206
DNA-GPT [58] ◇	0.5886	0.5913	0.1984	0.0171	0.4660	0.5021	0.0321	0.0330	0.4124	0.5010	0.0321	0.0227
NPR [46] ◇	0.5792	0.6068	0.2674	0.0050	0.4648	0.5155	0.0746	0.0361	0.4197	0.5051	0.0529	0.0216
DetectGPT [34] ◇	0.6314	0.6364	0.3132	0.0090	0.4871	0.5201	0.0631	0.0351	0.4371	0.5108	0.0601	0.0227
Fast-DetectGPT [7]	0.8419	0.7683	0.5402	0.3952	0.5251	0.5340	0.0746	0.0515	0.4958	0.5082	0.0168	0.0319
ImBD [9] ♠	0.8971	0.8260	0.6672	0.4253	0.5992	0.5758	0.1990	0.1732	0.6134	0.5824	0.1932	0.1586
DetectAnyLLM(ours) ♠	**0.9503**	**0.9032**	**0.8079**	**0.7743**	**0.9161**	**0.8619**	**0.7256**	**0.7577**	**0.9007**	**0.8435**	**0.6980**	**0.7168**
Imp.	+51.70%	+44.38%	+42.28%	+60.73%	+79.08%	+67.44%	+65.74%	+70.70%	+73.57%	+61.86%	+62.47%	+66.34%

MIRAGE-SIG, GPT-4o

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.4930	0.5427	0.0975	0.0040	0.4766	0.5046	0.0095	0.0278	0.4275	0.5000	0.0000	0.0103
LogRank [25]	0.4919	0.5407	0.0919	0.0030	0.4647	0.5000	0.0000	0.0247	0.4215	0.5000	0.0000	0.0103
Entropy [14]	0.6475	0.6246	0.2919	0.1176	0.5370	0.5391	0.0894	0.0525	0.5775	0.5622	0.1756	0.1080
RoBERTa-Base [32] ♠	0.4772	0.5106	0.0695	0.0583	0.5040	0.5113	0.0296	0.0443	0.5395	0.5273	0.0690	0.0720
RoBERTa-Large [32] ♠	0.4438	0.5010	0.0317	0.0221	0.5706	0.5608	0.1218	0.0731	0.6172	0.5844	0.1701	0.0874
LRR [46]	0.5006	0.5216	0.0486	0.0161	0.4272	0.5000	0.0000	0.0278	0.4087	0.5000	0.0000	0.0113
DNA-GPT [58] ◇	0.5610	0.5704	0.1482	0.0161	0.4732	0.5005	0.0227	0.0319	0.4222	0.5000	0.0000	0.0226
NPR [46] ◇	0.5619	0.5950	0.2502	0.0080	0.4716	0.5108	0.0647	0.0340	0.4297	0.5031	0.0276	0.0154
DetectGPT [34] ◇	0.5993	0.6171	0.2794	0.0040	0.4890	0.5263	0.0751	0.0247	0.4449	0.5087	0.0442	0.0113
Fast-DetectGPT [7]	0.8266	0.7487	0.4976	0.3588	0.5279	0.5371	0.0742	0.0566	0.4872	0.5062	0.0256	0.0267
ImBD [9] ♠	0.9048	0.8302	0.6690	0.4844	0.6118	0.5860	0.1947	0.1565	0.6120	0.5802	0.1726	0.1235
DetectAnyLLM(ours) ♠	**0.9656**	**0.9382**	**0.8783**	**0.8523**	**0.9080**	**0.8615**	**0.7268**	**0.7353**	**0.8697**	**0.8138**	**0.6410**	**0.6235**
Imp.	+63.87%	+63.61%	+63.22%	+71.35%	+76.29%	+66.54%	+66.07%	+68.62%	+65.95%	+55.20%	+56.45%	+57.04%

MIRAGE-DIG, GPT-4o-mini

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.5124	0.5523	0.1196	0.0060	0.4839	0.5202	0.0439	0.0174	0.4618	0.5131	0.0285	0.0179
LogRank [25]	0.5194	0.5513	0.1037	0.0090	0.4720	0.5093	0.0194	0.0153	0.4518	0.5053	0.0111	0.0137
Entropy [14]	0.6334	0.6226	0.2698	0.1186	0.5364	0.5447	0.1944	0.1242	0.5398	0.5426	0.1592	0.0988
RoBERTa-Base [32] ♠	0.5024	0.5256	0.1066	0.0995	0.4728	0.5027	0.0194	0.0490	0.5224	0.5168	0.0418	0.0557
RoBERTa-Large [32] ♠	0.5235	0.5176	0.0352	0.0372	0.5811	0.5550	0.1104	0.0926	0.6440	0.6041	0.2106	0.1094
LRR [46]	0.5377	0.5503	0.1005	0.0302	0.4252	0.5000	0.0000	0.0218	0.4190	0.5005	0.0132	0.0158
DNA-GPT [58] ◇	0.5846	0.5889	0.1964	0.0452	0.4770	0.5005	0.0135	0.0218	0.4469	0.5005	0.0229	0.0221
NPR [46] ◇	0.6006	0.6231	0.3056	0.0121	0.4904	0.5441	0.1212	0.0251	0.4261	0.5053	0.0434	0.0252
DetectGPT [34] ◇	0.6327	0.6402	0.3115	0.0161	0.5308	0.5572	0.1533	0.0370	0.4461	0.5142	0.0549	0.0252
Fast-DetectGPT [7]	0.8545	0.7779	0.5599	0.4362	0.5857	0.5752	0.1512	0.0643	0.5111	0.5189	0.0556	0.0336
ImBD [9] ♠	0.9101	0.8322	0.6677	0.5196	0.6767	0.6340	0.3098	0.2440	0.6267	0.6004	0.2265	0.1556
DetectAnyLLM(ours) ♠	**0.9611**	**0.9166**	**0.8336**	**0.8221**	**0.9496**	**0.9009**	**0.8022**	**0.8290**	**0.9209**	**0.8617**	**0.7248**	**0.7350**
Imp.	+56.74%	+50.30%	+49.92%	+62.97%	+84.40%	+72.92%	+71.35%	+77.38%	+77.78%	+65.07%	+64.43%	+68.62%

MIRAGE-SIG, GPT-4o-mini

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.5099	0.5592	0.1241	0.0110	0.5119	0.5342	0.0725	0.0271	0.4555	0.5000	0.0000	0.0115
LogRank [25]	0.5103	0.5527	0.1134	0.0100	0.4975	0.5212	0.0428	0.0261	0.4432	0.5000	0.0000	0.0105
Entropy [14]	0.6334	0.6108	0.2675	0.1304	0.5117	0.5364	0.1659	0.1021	0.5447	0.5387	0.1808	0.1067
RoBERTa-Base [32] ♠	0.5031	0.5221	0.0693	0.0672	0.4734	0.5000	0.0000	0.0304	0.5338	0.5214	0.0614	0.0680
RoBERTa-Large [32] ♠	0.5418	0.5286	0.0614	0.0692	0.5808	0.5679	0.1361	0.0803	0.6331	0.5973	0.1967	0.0690
LRR [46]	0.5189	0.5391	0.0784	0.0271	0.4432	0.5000	0.0000	0.0185	0.4048	0.5000	0.0000	0.0146
DNA-GPT [58] ◇	0.5687	0.5752	0.1552	0.0321	0.4971	0.5195	0.0425	0.0337	0.4401	0.5000	0.0000	0.0199
NPR [46] ◇	0.5882	0.6128	0.2591	0.0070	0.4868	0.5206	0.0724	0.0358	0.4274	0.5078	0.0542	0.0230
DetectGPT [34] ◇	0.6199	0.6249	0.2804	0.0070	0.5303	0.5451	0.1058	0.0271	0.4446	0.5126	0.0606	0.0251
Fast-DetectGPT [7]	0.8398	0.7658	0.5323	0.3731	0.5860	0.5717	0.1435	0.0749	0.4960	0.5136	0.0422	0.0356
ImBD [9] ♠	0.9158	0.8340	0.6695	0.5637	0.6629	0.6260	0.2812	0.2356	0.6251	0.5988	0.2279	0.1799
DetectAnyLLM(ours) ♠	**0.9656**	**0.9418**	**0.8847**	**0.8365**	**0.9425**	**0.8925**	**0.7861**	**0.8089**	**0.9176**	**0.8708**	**0.7467**	**0.7416**
Imp.	+59.18%	+64.95%	+65.11%	+62.53%	+82.93%	+71.26%	+70.24%	+75.00%	+77.54%	+67.80%	+67.19%	+68.49%

Table 16: Generator: GPT-o3-mini, Moonshot-v1. "Imp.": Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$.

MIRAGE-DIG, GPT-o3-mini

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.2513	0.5015	0.0317	0.0030	0.4473	0.5022	0.0331	0.0164	0.4064	0.5005	0.0232	0.0086
LogRank [25]	0.2465	0.5010	0.0317	0.0030	0.4319	0.5005	0.0135	0.0120	0.3951	0.5000	0.0000	0.0065
Entropy [14]	0.7324	0.6819	0.3674	0.1573	0.5222	0.5377	0.1187	0.0350	0.5584	0.5500	0.1343	0.1000
RoBERTa-Base [32] ♠	0.4050	0.5000	0.0000	0.0080	0.4395	0.5000	0.0000	0.0175	0.4779	0.5005	0.0056	0.0376
RoBERTa-Large [32] ♠	0.4319	0.5005	0.0224	0.0271	0.5169	0.5055	0.0111	0.0438	0.5680	0.5457	0.0945	0.0742
LRR [46]	0.2687	0.5010	0.0317	0.0040	0.3927	0.5000	0.0000	0.0120	0.3628	0.5000	0.0000	0.0226
DNA-GPT [58] ♢	0.2675	0.5005	0.0068	0.0020	0.4441	0.5038	0.0424	0.0164	0.4124	0.5005	0.0232	0.0129
NPR [46] ♢	0.3819	0.5205	0.1046	0.0040	0.4497	0.5153	0.0604	0.0241	0.3986	0.5048	0.0541	0.0118
DetectGPT [34] ♢	0.3953	0.5210	0.1004	0.0040	0.4720	0.5252	0.0932	0.0263	0.4167	0.5065	0.0514	0.0172
Fast-DetectGPT [7]	0.4107	0.5005	0.0068	0.0291	0.4327	0.5000	0.0000	0.0252	0.3981	0.5000	0.0000	0.0140
ImBD [9] ♠	0.8792	0.8151	0.6468	0.2956	0.6438	0.6138	0.2550	0.1805	0.6563	0.6199	0.2652	0.2140
DetectAnyLLM(ours) ♠	**0.9368**	**0.8993**	**0.7995**	**0.6162**	**0.9008**	**0.8414**	**0.6858**	**0.6751**	**0.8987**	**0.8484**	**0.6986**	**0.7118**
Imp.	+47.68%	+45.53%	+43.24%	+45.52%	+72.15%	+58.92%	+57.82%	+60.35%	+70.53%	+60.11%	+58.99%	+63.34%

MIRAGE-SIG, GPT-o3-mini

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.2405	0.5010	0.0142	0.0000	0.4470	0.5011	0.0232	0.0238	0.4148	0.5000	0.0000	0.0054
LogRank [25]	0.2361	0.5000	0.0000	0.0000	0.4304	0.5005	0.0232	0.0173	0.4039	0.5000	0.0000	0.0064
Entropy [14]	0.7380	0.6795	0.3937	0.1645	0.5294	0.5394	0.0925	0.0454	0.5581	0.5494	0.1298	0.0806
RoBERTa-Base [32] ♠	0.3946	0.5000	0.0000	0.0090	0.4393	0.5000	0.0000	0.0173	0.4689	0.5000	0.0000	0.0333
RoBERTa-Large [32] ♠	0.4466	0.5000	0.0000	0.0221	0.4968	0.5022	0.0093	0.0454	0.5516	0.5387	0.0774	0.0494
LRR [46]	0.2620	0.5000	0.0000	0.0020	0.3888	0.5000	0.0000	0.0194	0.3763	0.5000	0.0000	0.0172
DNA-GPT [58] ♢	0.2672	0.5000	0.0000	0.0000	0.4508	0.5032	0.0232	0.0216	0.4145	0.5005	0.0232	0.0215
NPR [46] ♢	0.3966	0.5110	0.0486	0.0030	0.4536	0.5124	0.0795	0.0270	0.4089	0.5027	0.0439	0.0161
DetectGPT [34] ♢	0.4016	0.5120	0.0341	0.0030	0.4784	0.5205	0.0782	0.0259	0.4321	0.5016	0.0190	0.0183
Fast-DetectGPT [7]	0.4093	0.5000	0.0000	0.0271	0.4400	0.5005	0.0065	0.0205	0.4139	0.5000	0.0000	0.0204
ImBD [9] ♠	0.8963	0.8340	0.6762	0.3942	0.6432	0.6220	0.2589	0.1901	0.6429	0.6139	0.2372	0.1923
DetectAnyLLM(ours) ♠	**0.9528**	**0.9293**	**0.8598**	**0.7773**	**0.8980**	**0.8499**	**0.7008**	**0.6976**	**0.8948**	**0.8528**	**0.7110**	**0.7143**
Imp.	+54.52%	+57.40%	+56.70%	+63.25%	+71.40%	+60.29%	+59.63%	+62.67%	+70.54%	+61.89%	+62.12%	+64.63%

MIRAGE-DIG, Moonshot-v1

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.6179	0.6155	0.2547	0.0307	0.5356	0.5565	0.1146	0.0337	0.4927	0.5189	0.0482	0.0263
LogRank [25]	0.6386	0.6207	0.2632	0.0419	0.5273	0.5457	0.0958	0.0348	0.4838	0.5084	0.0173	0.0242
Entropy [14]	0.6193	0.6017	0.2364	0.0532	0.5239	0.5348	0.0839	0.0652	0.5430	0.5425	0.1129	0.0683
RoBERTa-Base [32] ♠	0.6624	0.6334	0.2722	0.2117	0.5236	0.5261	0.0849	0.0891	0.5427	0.5357	0.0924	0.0945
RoBERTa-Large [32] ♠	0.5987	0.5721	0.1670	0.1380	0.5838	0.5679	0.1462	0.1087	0.6118	0.5788	0.1685	0.1355
LRR [46]	0.6878	0.6380	0.2815	0.1237	0.4909	0.5098	0.0196	0.0293	0.4549	0.5000	0.0000	0.0252
DNA-GPT [58] ♢	0.7157	0.6621	0.3426	0.1820	0.5525	0.5549	0.1098	0.0370	0.4995	0.5105	0.0586	0.0462
NPR [46] ♢	0.6852	0.6682	0.3636	0.0215	0.5469	0.5668	0.1551	0.0467	0.4807	0.5189	0.0708	0.0315
DetectGPT [34] ♢	0.6783	0.6595	0.3531	0.0112	0.5860	0.5859	0.1946	0.0511	0.5108	0.5331	0.1038	0.0294
Fast-DetectGPT [7]	**0.9069**	0.8298	0.6603	**0.6319**	0.6829	0.6315	0.2815	0.1837	0.6073	0.5814	0.1842	0.1429
ImBD [9] ♠	0.7824	0.7014	0.4051	0.3405	0.6475	0.6141	0.2693	0.2185	0.6206	0.5930	0.2280	0.1912
DetectAnyLLM(ours) ♠	0.9057	**0.8497**	**0.6996**	0.4468	**0.9243**	**0.8652**	**0.7308**	**0.7163**	**0.9047**	**0.8503**	**0.7036**	**0.7258**
Imp.	–	+11.71%	+11.55%	–	+76.14%	+63.42%	+62.53%	+63.70%	+74.88%	+63.23%	+61.60%	+66.10%

MIRAGE-SIG, Moonshot-v1

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.6159	0.6264	0.2707	0.0091	0.5484	0.5562	0.1211	0.0375	0.4993	0.5253	0.0512	0.0169
LogRank [25]	0.6344	0.6244	0.2787	0.0152	0.5372	0.5469	0.1007	0.0331	0.4921	0.5205	0.0411	0.0190
Entropy [14]	0.6177	0.6081	0.2559	0.0548	0.5194	0.5281	0.1102	0.0628	0.5482	0.5511	0.1297	0.0801
RoBERTa-Base [32] ♠	0.6504	0.6223	0.2690	0.1939	0.5128	0.5204	0.0582	0.0706	0.5534	0.5474	0.1178	0.0927
RoBERTa-Large [32] ♠	0.6190	0.5898	0.1871	0.1492	0.5865	0.5656	0.1316	0.0970	0.6171	0.5864	0.1730	0.1106
LRR [46]	0.6810	0.6411	0.2853	0.0975	0.4878	0.5061	0.0235	0.0320	0.4638	0.5000	0.0000	0.0253
DNA-GPT [58] ♢	0.7070	0.6640	0.3327	0.1076	0.5564	0.5507	0.1029	0.0562	0.5271	0.5295	0.0774	0.0390
NPR [46] ♢	0.6649	0.6629	0.3389	0.0254	0.5484	0.5579	0.1314	0.0386	0.4985	0.5353	0.0954	0.0221
DetectGPT [34] ♢	0.6589	0.6492	0.3354	0.0173	0.5872	0.5877	0.1778	0.0430	0.5317	0.5421	0.1267	0.0358
Fast-DetectGPT [7]	0.9113	0.8406	0.6813	0.6254	0.6958	0.6527	0.3131	0.1996	0.6369	0.6001	0.2072	0.1444
ImBD [9] ♠	0.7529	0.6716	0.3626	0.3178	0.6675	0.6356	0.2861	0.2348	0.6221	0.5948	0.2489	0.2192
DetectAnyLLM(ours) ♠	**0.9421**	**0.9056**	**0.8113**	**0.7015**	**0.9248**	**0.8682**	**0.7406**	**0.7497**	**0.8988**	**0.8361**	**0.6795**	**0.6786**
Imp.	+34.72%	+40.76%	+40.81%	+20.33%	+75.27%	+62.06%	+62.23%	+67.29%	+72.12%	+59.03%	+57.33%	+58.84%

Table 17: Generator: DeepSeek-R1, DeepSeek-V3. "Imp.": Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$.

MIRAGE-DIG, DeepSeek-R1

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.3248	0.5005	0.0224	0.0010	0.3533	0.5000	0.0000	0.0100	0.3285	0.5000	0.0000	0.0093
LogRank [25]	0.2896	0.5005	0.0224	0.0010	0.3235	0.5000	0.0000	0.0089	0.3037	0.5000	0.0000	0.0062
Entropy [14]	0.6454	0.6242	0.2939	0.0902	0.5593	0.5456	0.1194	0.1000	0.5772	0.5542	0.1638	0.1373
RoBERTa-Base [32] ♠	0.5335	0.5000	0.0000	0.0000	0.4483	0.5000	0.0000	0.0000	0.4207	0.5000	0.0000	0.0021
RoBERTa-Large [32] ♠	0.2018	0.5000	0.0000	0.0000	0.2631	0.5000	0.0000	0.0056	0.3501	0.5000	0.0000	0.0196
LRR [46]	0.2159	0.5005	0.0224	0.0000	0.2493	0.5000	0.0000	0.0089	0.2508	0.5000	0.0000	0.0072
DNA-GPT [58] ◇	0.2842	0.5025	0.0448	0.0000	0.3112	0.5000	0.0000	0.0100	0.2962	0.5005	0.0227	0.0114
NPR [46] ◇	0.5706	0.6172	0.2931	0.0130	0.5089	0.5556	0.1648	0.0267	0.4591	0.5253	0.0775	0.0175
DetectGPT [34] ◇	0.6776	0.6623	0.3693	0.0451	0.5792	0.5811	0.2026	0.0400	0.5111	0.5320	0.0912	0.0248
Fast-DetectGPT [7]	0.3838	0.5000	0.0000	0.0230	0.2661	0.5000	0.0000	0.0111	0.2360	0.5000	0.0000	0.0041
ImBD [9] ♠	0.8972	0.8181	0.6363	0.5020	0.7989	0.7333	0.4855	0.4044	0.7307	0.6847	0.3779	0.3189
DetectAnyLLM(ours) ♠	**0.9566**	**0.9088**	**0.8194**	**0.7505**	**0.9720**	**0.9278**	**0.8556**	**0.8978**	**0.9612**	**0.9262**	**0.8532**	**0.8958**
Imp.	+57.78%	+49.86%	+50.36%	+49.90%	+86.09%	+72.92%	+71.93%	+82.84%	+85.58%	+76.60%	+76.40%	+84.70%

MIRAGE-SIG, DeepSeek-R1

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.3301	0.5015	0.0224	0.0040	0.3358	0.5006	0.0235	0.0122	0.3347	0.5000	0.0000	0.0083
LogRank [25]	0.2983	0.5000	0.0000	0.0040	0.3063	0.5000	0.0000	0.0110	0.3098	0.5000	0.0000	0.0072
Entropy [14]	0.6389	0.6048	0.2534	0.1284	0.5739	0.5475	0.1313	0.1094	0.5759	0.5537	0.1895	0.1259
RoBERTa-Base [32] ♠	0.5245	0.5000	0.0000	0.0000	0.4373	0.5000	0.0000	0.0000	0.4201	0.5000	0.0000	0.0010
RoBERTa-Large [32] ♠	0.2007	0.5000	0.0000	0.0000	0.2768	0.5000	0.0000	0.0066	0.3504	0.5005	0.0227	0.0124
LRR [46]	0.2322	0.5000	0.0000	0.0020	0.2396	0.5000	0.0000	0.0055	0.2483	0.5000	0.0000	0.0052
DNA-GPT [58] ◇	0.2830	0.5010	0.0100	0.0000	0.2896	0.5000	0.0000	0.0033	0.2982	0.5000	0.0000	0.0062
NPR [46] ◇	0.5762	0.6138	0.2804	0.0181	0.5115	0.5403	0.1516	0.0221	0.4631	0.5263	0.0867	0.0175
DetectGPT [34] ◇	0.6823	0.6605	0.3485	0.0451	0.5821	0.5751	0.1658	0.0309	0.5135	0.5335	0.1215	0.0351
Fast-DetectGPT [7]	0.3874	0.5000	0.0000	0.0261	0.2580	0.5000	0.0000	0.0088	0.2437	0.5000	0.0000	0.0031
ImBD [9] ♠	0.9087	0.8400	0.6803	0.4835	0.7885	0.7249	0.4693	0.4320	0.7383	0.6827	0.3754	0.3148
DetectAnyLLM(ours) ♠	**0.9617**	**0.9303**	**0.8617**	**0.8646**	**0.9696**	**0.9215**	**0.8437**	**0.8796**	**0.9534**	**0.9174**	**0.8352**	**0.8772**
Imp.	+58.04%	+56.43%	+56.74%	+73.79%	+85.63%	+71.49%	+70.55%	+78.79%	+82.19%	+73.98%	+73.61%	+82.08%

MIRAGE-DIG, DeepSeek-V3

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.6430	0.6484	0.3038	0.0201	0.6044	0.5885	0.1849	0.0421	0.5370	0.5610	0.1292	0.0242
LogRank [25]	0.6495	0.6439	0.2991	0.0272	0.5893	0.5785	0.1683	0.0432	0.5242	0.5547	0.1136	0.0231
Entropy [14]	0.5548	0.5805	0.2026	0.0523	0.4305	0.5079	0.0401	0.0285	0.4848	0.5258	0.1164	0.0852
RoBERTa-Base [32] ♠	0.5416	0.5302	0.1092	0.1026	0.4806	0.5000	0.0000	0.0316	0.4764	0.5005	0.0050	0.0410
RoBERTa-Large [32] ♠	0.4194	0.5015	0.0389	0.0322	0.5217	0.5216	0.0439	0.0421	0.5530	0.5373	0.0769	0.0641
LRR [46]	0.6567	0.6363	0.2800	0.0744	0.5189	0.5269	0.0545	0.0443	0.4758	0.5095	0.0206	0.0263
DNA-GPT [58] ◇	0.6980	0.6635	0.3314	0.0885	0.5620	0.5558	0.1308	0.0527	0.5146	0.5310	0.0667	0.0347
NPR [46] ◇	0.7026	0.6977	0.4364	0.0292	0.5712	0.5732	0.1820	0.0558	0.5230	0.5568	0.1394	0.0358
DetectGPT [34] ◇	0.7581	0.7294	0.4737	0.0412	0.6084	0.5959	0.2169	0.0601	0.5656	0.5773	0.1914	0.0379
Fast-DetectGPT [7]	0.9415	0.8732	0.7481	**0.7435**	0.6353	0.6064	0.2148	0.1191	0.6088	0.5910	0.1871	0.0715
ImBD [9] ♠	0.9119	0.8270	0.6550	0.5765	0.6800	0.6444	0.3130	0.2561	0.6930	0.6598	0.3373	0.3028
DetectAnyLLM(ours) ♠	**0.9473**	**0.8999**	**0.8016**	0.7404	**0.9152**	**0.8541**	**0.7122**	**0.7208**	**0.9125**	**0.8649**	**0.7353**	**0.7497**
Imp.	+9.97%	+21.03%	+21.24%	–	+73.49%	+58.96%	+58.10%	+62.46%	+71.51%	+60.28%	+60.06%	+64.10%

MIRAGE-SIG, DeepSeek-V3

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.6522	0.6505	0.3055	0.0291	0.6144	0.6056	0.2211	0.0442	0.5545	0.5665	0.1371	0.0211
LogRank [25]	0.6556	0.6454	0.2944	0.0321	0.5980	0.5911	0.1826	0.0453	0.5407	0.5533	0.1066	0.0211
Entropy [14]	0.5455	0.5682	0.1772	0.0532	0.4244	0.5048	0.0388	0.0248	0.4753	0.5322	0.1078	0.0749
RoBERTa-Base [32] ♠	0.5305	0.5246	0.0891	0.0782	0.4630	0.5000	0.0000	0.0409	0.4912	0.5053	0.0386	0.0454
RoBERTa-Large [32] ♠	0.4179	0.5005	0.0224	0.0341	0.5121	0.5086	0.0430	0.0603	0.5509	0.5438	0.0882	0.0538
LRR [46]	0.6556	0.6249	0.2576	0.0612	0.5252	0.5318	0.0641	0.0442	0.4839	0.5084	0.0230	0.0232
DNA-GPT [58] ◇	0.7149	0.6760	0.3687	0.0802	0.5779	0.5722	0.1516	0.0420	0.5239	0.5311	0.0786	0.0359
NPR [46] ◇	0.6938	0.6871	0.4142	0.0281	0.5728	0.5781	0.1673	0.0442	0.5397	0.5570	0.1385	0.0253
DetectGPT [34] ◇	0.7579	0.7197	0.4701	0.0381	0.6136	0.6067	0.2236	0.0506	0.5783	0.5770	0.1724	0.0348
Fast-DetectGPT [7]	0.9359	0.8666	0.7332	0.6971	0.6385	0.6110	0.2237	0.1196	0.6162	0.5833	0.1772	0.1086
ImBD [9] ♠	0.9177	0.8360	0.6728	0.6018	0.6868	0.6439	0.3127	0.2737	0.6880	0.6572	0.3267	0.2985
DetectAnyLLM(ours) ♠	**0.9656**	**0.9343**	**0.8687**	**0.8887**	**0.9286**	**0.8723**	**0.7448**	**0.7392**	**0.9112**	**0.8660**	**0.7329**	**0.7373**
Imp.	+46.32%	+50.75%	+50.80%	+63.25%	+77.21%	+64.15%	+62.87%	+64.09%	+71.54%	+60.92%	+60.33%	+62.56%

Table 18: Generator: Claude-3.5-Haiku, 3.7-sonnet. "Imp.": Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$.

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
MIRAGE-DIG, Claude-3.5-haiku												
Likelihood [44]	0.3121	0.5000	0.0000	0.0000	0.2962	0.5000	0.0000	0.0000	0.2763	0.5000	0.0000	0.0011
LogRank [25]	0.3153	0.5000	0.0000	0.0000	0.2761	0.5000	0.0000	0.0000	0.2592	0.5000	0.0000	0.0011
Entropy [14]	0.7963	0.7320	0.4736	0.2660	0.7422	0.6918	0.3840	0.1875	0.7518	0.6762	0.3784	0.2338
RoBERTa-Base [32] ♠	0.5036	0.5090	0.0545	0.0510	0.4503	0.5000	0.0000	0.0098	0.4400	0.5000	0.0000	0.0103
RoBERTa-Large [32] ♠	0.4095	0.5005	0.0049	0.0370	0.3540	0.5000	0.0000	0.0110	0.3947	0.5000	0.0000	0.0194
LRR [46]	0.3547	0.5000	0.0000	0.0060	0.2417	0.5000	0.0000	0.0000	0.2319	0.5000	0.0000	0.0023
DNA-GPT [58] ♢	0.4682	0.5020	0.0259	0.0120	0.4072	0.5012	0.0248	0.0086	0.3809	0.5000	0.0000	0.0080
NPR [46] ♢	0.5282	0.5800	0.2244	0.0000	0.4723	0.5484	0.1407	0.0061	0.4520	0.5496	0.1446	0.0034
DetectGPT [34] ♢	0.5598	0.5870	0.2290	0.0050	0.5418	0.5705	0.1724	0.0196	0.5318	0.5661	0.1612	0.0171
Fast-DetectGPT [7]	0.7517	0.6875	0.3768	0.2540	0.6380	0.6134	0.2292	0.1005	0.6291	0.5992	0.2061	0.1163
ImBD [9] ♠	0.9153	0.8460	0.7042	0.5010	0.8748	0.7843	0.5709	0.5061	0.8554	0.7754	0.5510	0.5245
DetectAnyLLM(ours) ♠	**0.9441**	**0.9090**	**0.8195**	**0.6650**	**0.9903**	**0.9602**	**0.9203**	**0.9681**	**0.9796**	**0.9458**	**0.8917**	**0.9396**
Imp.	+33.97%	+40.91%	+38.96%	+32.87%	+92.24%	+81.53%	+81.44%	+93.55%	+85.87%	+75.89%	+75.88%	+87.29%
MIRAGE-SIG, Claude-3.5-haiku												
Likelihood [44]	0.3260	0.5000	0.0000	0.0000	0.2745	0.5000	0.0000	0.0012	0.2880	0.5000	0.0000	0.0023
LogRank [25]	0.3267	0.5000	0.0000	0.0000	0.2574	0.5000	0.0000	0.0012	0.2738	0.5000	0.0000	0.0023
Entropy [14]	0.7826	0.7210	0.4428	0.2216	0.7617	0.6951	0.3924	0.2122	0.7445	0.6680	0.3708	0.2526
RoBERTa-Base [32] ♠	0.4608	0.5035	0.0316	0.0282	0.4257	0.5000	0.0000	0.0024	0.4356	0.5000	0.0000	0.0137
RoBERTa-Large [32] ♠	0.4108	0.5000	0.0000	0.0222	0.3683	0.5000	0.0000	0.0085	0.4150	0.5006	0.0239	0.0126
LRR [46]	0.3568	0.5000	0.0000	0.0010	0.2302	0.5000	0.0000	0.0000	0.2540	0.5000	0.0000	0.0023
DNA-GPT [58] ♢	0.4659	0.5146	0.0597	0.0081	0.3960	0.5006	0.0247	0.0085	0.3767	0.5000	0.0000	0.0080
NPR [46] ♢	0.5427	0.5886	0.2348	0.0010	0.4712	0.5433	0.1601	0.0061	0.4507	0.5440	0.1299	0.0046
DetectGPT [34] ♢	0.5757	0.5972	0.2481	0.0000	0.5255	0.5567	0.1426	0.0110	0.5317	0.5577	0.1278	0.0103
Fast-DetectGPT [7]	0.7419	0.6757	0.3529	0.2346	0.6195	0.5921	0.1842	0.1256	0.6152	0.5954	0.1983	0.1326
ImBD [9] ♠	0.9223	0.8550	0.7167	0.5257	0.8808	0.7909	0.5835	0.5451	0.8489	0.7691	0.5430	0.4651
DetectAnyLLM(ours) ♠	**0.9621**	**0.9421**	**0.8858**	**0.8540**	**0.9844**	**0.9433**	**0.8866**	**0.9354**	**0.9759**	**0.9429**	**0.8859**	**0.9326**
Imp.	+51.26%	+60.07%	+59.67%	+69.21%	+86.93%	+72.89%	+72.78%	+85.79%	+84.01%	+75.25%	+75.04%	+87.39%
MIRAGE-DIG, Claude-3.7-sonnet												
Likelihood [44]	0.2908	0.5015	0.0317	0.0020	0.3740	0.5000	0.0000	0.0157	0.3061	0.5000	0.0000	0.0083
LogRank [25]	0.2868	0.5010	0.0224	0.0020	0.3619	0.5000	0.0000	0.0168	0.2939	0.5000	0.0000	0.0062
Entropy [14]	0.7081	0.6633	0.3289	0.1528	0.5848	0.5613	0.1928	0.1342	0.6360	0.5937	0.2416	0.1905
RoBERTa-Base [32] ♠	0.4216	0.5000	0.0000	0.0040	0.4425	0.5000	0.0000	0.0126	0.4188	0.5000	0.0000	0.0072
RoBERTa-Large [32] ♠	0.2314	0.5000	0.0000	0.0010	0.3875	0.5000	0.0000	0.0115	0.4109	0.5005	0.0228	0.0280
LRR [46]	0.2995	0.5020	0.0449	0.0060	0.3310	0.5000	0.0000	0.0168	0.2739	0.5000	0.0000	0.0114
DNA-GPT [58] ♢	0.3649	0.5000	0.0000	0.0020	0.4039	0.5000	0.0000	0.0220	0.3407	0.5005	0.0228	0.0197
NPR [46] ♢	0.5223	0.5698	0.1927	0.0070	0.4563	0.5168	0.0630	0.0377	0.4098	0.5036	0.0321	0.0155
DetectGPT [34] ♢	0.5435	0.5754	0.2201	0.0151	0.4833	0.5168	0.0708	0.0283	0.4364	0.5062	0.0443	0.0186
Fast-DetectGPT [7]	0.4048	0.5000	0.0000	0.0211	0.3466	0.5005	0.0229	0.0168	0.2992	0.5000	0.0000	0.0104
ImBD [9] ♠	**0.8576**	0.7920	0.5887	**0.3307**	0.6024	0.5755	0.1640	0.1111	0.6319	0.6020	0.2070	0.1356
DetectAnyLLM(ours) ♠	0.8526	**0.8015**	**0.6057**	0.3136	**0.9096**	**0.8538**	**0.7114**	**0.7201**	**0.9167**	**0.8732**	**0.7483**	**0.7816**
Imp.	–	+4.59%	+4.13%	–	+77.27%	+65.56%	+64.24%	+67.68%	+77.12%	+68.14%	+66.81%	+73.02%
MIRAGE-SIG, Claude-3.7-sonnet												
Likelihood [44]	0.3055	0.5000	0.0000	0.0070	0.3663	0.5000	0.0000	0.0178	0.3158	0.5000	0.0000	0.0072
LogRank [25]	0.2986	0.5000	0.0000	0.0050	0.3541	0.5000	0.0000	0.0167	0.3068	0.5000	0.0000	0.0072
Entropy [14]	0.6916	0.6406	0.2926	0.1491	0.5834	0.5575	0.2005	0.1233	0.6278	0.5916	0.2277	0.1543
RoBERTa-Base [32] ♠	0.4125	0.5000	0.0000	0.0020	0.4220	0.5000	0.0000	0.0094	0.4332	0.5000	0.0000	0.0144
RoBERTa-Large [32] ♠	0.2389	0.5000	0.0000	0.0010	0.3955	0.5000	0.0000	0.0261	0.4182	0.5000	0.0000	0.0154
LRR [46]	0.3036	0.5000	0.0000	0.0050	0.3277	0.5000	0.0000	0.0104	0.2974	0.5000	0.0000	0.0093
DNA-GPT [58] ♢	0.3769	0.5000	0.0000	0.0110	0.3917	0.5000	0.0000	0.0230	0.3536	0.5005	0.0227	0.0134
NPR [46] ♢	0.5168	0.5616	0.1623	0.0100	0.4607	0.5094	0.0606	0.0282	0.4104	0.5093	0.0394	0.0103
DetectGPT [34] ♢	0.5366	0.5591	0.1794	0.0180	0.4880	0.5094	0.0433	0.0261	0.4330	0.5041	0.0227	0.0144
Fast-DetectGPT [7]	0.4086	0.5000	0.0000	0.0270	0.3396	0.5000	0.0000	0.0167	0.2944	0.5000	0.0000	0.0113
ImBD [9] ♠	0.8568	0.7858	0.5739	0.3103	0.6237	0.6082	0.2188	0.1160	0.6113	0.5921	0.1842	0.1152
DetectAnyLLM(ours) ♠	**0.8836**	**0.8478**	**0.6966**	**0.4565**	**0.9151**	**0.8600**	**0.7233**	**0.7429**	**0.9102**	**0.8735**	**0.7487**	**0.7623**
Imp.	+18.71%	+28.97%	+28.79%	+21.19%	+77.43%	+64.27%	+64.58%	+70.68%	+75.89%	+68.98%	+67.46%	+71.90%

Table 19: Generator: Gemini-2.0-flash, flash-lite. "Imp.": Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$.

MIRAGE-DIG, Gemini-2.0-flash

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.4471	0.5020	0.0052	0.0193	0.4069	0.5000	0.0000	0.0197	0.3967	0.5000	0.0000	0.0054
LogRank [25]	0.4445	0.5000	0.0000	0.0234	0.3876	0.5000	0.0000	0.0219	0.3837	0.5000	0.0000	0.0075
Entropy [14]	0.6913	0.6514	0.3071	0.1626	0.6132	0.5864	0.2002	0.1488	0.6299	0.5927	0.2471	0.1726
RoBERTa-Base [32] ♠	0.4250	0.5010	0.0101	0.0274	0.4100	0.5000	0.0000	0.0175	0.4409	0.5000	0.0000	0.0214
RoBERTa-Large [32] ♠	0.2978	0.5005	0.0225	0.0091	0.4322	0.5000	0.0000	0.0208	0.4876	0.5032	0.0232	0.0482
LRR [46]	0.4521	0.5005	0.0225	0.0386	0.3392	0.5000	0.0000	0.0066	0.3523	0.5000	0.0000	0.0086
DNA-GPT [58] ◇	0.6123	0.5971	0.2079	0.0447	0.4558	0.5005	0.0234	0.0295	0.4328	0.5005	0.0232	0.0193
NPR [46] ◇	0.6319	0.6418	0.3309	0.0224	0.5142	0.5454	0.1278	0.0252	0.4957	0.5461	0.1435	0.0182
DetectGPT [34] ◇	0.6715	0.6550	0.3516	0.0366	0.5671	0.5700	0.1670	0.0383	0.5530	0.5648	0.1620	0.0300
Fast-DetectGPT [7]	0.8157	0.7464	0.4929	0.3953	0.5862	0.5706	0.1426	0.0886	0.6026	0.5841	0.1703	0.1125
ImBD [9] ♠	0.8402	0.7597	0.5261	0.3313	0.6612	0.6329	0.2858	0.2243	0.6723	0.6372	0.3046	0.2669
DetectAnyLLM(ours) ♠	**0.9265**	**0.8775**	**0.7558**	**0.6514**	**0.9541**	**0.9114**	**0.8229**	**0.8490**	**0.9559**	**0.9148**	**0.8303**	**0.8767**
Imp.	+54.01%	+49.05%	+48.47%	+42.35%	+86.45%	+75.86%	+75.20%	+80.54%	+86.55%	+76.51%	+75.60%	+83.19%

MIRAGE-SIG, Gemini-2.0-flash

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.4480	0.5046	0.0107	0.0102	0.4047	0.5000	0.0000	0.0198	0.3812	0.5000	0.0000	0.0097
LogRank [25]	0.4457	0.5025	0.0058	0.0102	0.3892	0.5000	0.0000	0.0165	0.3689	0.5000	0.0000	0.0108
Entropy [14]	0.6923	0.6463	0.3137	0.1376	0.6229	0.5862	0.2435	0.1603	0.6453	0.6016	0.2574	0.1914
RoBERTa-Base [32] ♠	0.4079	0.5005	0.0045	0.0214	0.4098	0.5005	0.0234	0.0198	0.4455	0.5000	0.0000	0.0292
RoBERTa-Large [32] ♠	0.3014	0.5000	0.0000	0.0020	0.4495	0.5005	0.0234	0.0439	0.4829	0.5005	0.0233	0.0432
LRR [46]	0.4568	0.5000	0.0000	0.0204	0.3524	0.5000	0.0000	0.0132	0.3407	0.5000	0.0000	0.0119
DNA-GPT [58] ◇	0.6038	0.5897	0.1968	0.0449	0.4597	0.5000	0.0000	0.0263	0.4227	0.5000	0.0000	0.0205
NPR [46] ◇	0.6156	0.6346	0.2973	0.0224	0.5297	0.5543	0.1354	0.0263	0.4995	0.5481	0.1396	0.0086
DetectGPT [34] ◇	0.6589	0.6488	0.3231	0.0255	0.5820	0.5851	0.1887	0.0285	0.5567	0.5638	0.1656	0.0205
Fast-DetectGPT [7]	0.8090	0.7334	0.4756	0.3802	0.6004	0.5801	0.1604	0.1098	0.5872	0.5692	0.1493	0.1005
ImBD [9] ♠	0.8395	0.7604	0.5217	0.3578	0.6838	0.6454	0.3064	0.2437	0.6738	0.6308	0.2770	0.2097
DetectAnyLLM(ours) ♠	**0.9477**	**0.9139**	**0.8282**	**0.7676**	**0.9536**	**0.9034**	**0.8076**	**0.8419**	**0.9524**	**0.9103**	**0.8212**	**0.8595**
Imp.	+67.44%	+64.04%	+64.08%	+62.50%	+85.31%	+72.76%	+72.26%	+79.10%	+85.40%	+75.70%	+75.27%	+82.22%

MIRAGE-DIG, Gemini-2.0-flash-lite

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.4082	0.5005	0.0226	0.0041	0.3996	0.5000	0.0000	0.0185	0.3921	0.5000	0.0000	0.0128
LogRank [25]	0.4178	0.5000	0.0000	0.0082	0.3841	0.5000	0.0000	0.0131	0.3824	0.5000	0.0000	0.0117
Entropy [14]	0.7528	0.7028	0.4084	0.1869	0.6432	0.6009	0.2350	0.1778	0.6571	0.6102	0.2473	0.1864
RoBERTa-Base [32] ♠	0.4395	0.5005	0.0035	0.0429	0.4494	0.5000	0.0000	0.0262	0.4534	0.5000	0.0000	0.0330
RoBERTa-Large [32] ♠	0.3808	0.5000	0.0000	0.0184	0.4941	0.5049	0.0420	0.0393	0.5182	0.5106	0.0462	0.0437
LRR [46]	0.4634	0.5005	0.0226	0.0378	0.3462	0.5000	0.0000	0.0164	0.3636	0.5000	0.0000	0.0128
DNA-GPT [58] ◇	0.5756	0.5746	0.1520	0.0429	0.4564	0.5005	0.0234	0.0294	0.4320	0.5000	0.0000	0.0266
NPR [46] ◇	0.6043	0.6185	0.2616	0.0143	0.5133	0.5458	0.1394	0.0229	0.4865	0.5346	0.1249	0.0202
DetectGPT [34] ◇	0.6280	0.6318	0.2966	0.0174	0.5623	0.5671	0.1658	0.0164	0.5272	0.5506	0.1576	0.0202
Fast-DetectGPT [7]	0.8464	0.7712	0.5426	0.4331	0.6605	0.6309	0.2736	0.1679	0.6599	0.6235	0.2482	0.1502
ImBD [9] ♠	0.8604	0.7891	0.5792	0.3626	0.6965	0.6483	0.3153	0.2628	0.6706	0.6400	0.3057	0.2662
DetectAnyLLM(ours) ♠	**0.9122**	**0.8601**	**0.7204**	**0.5485**	**0.9690**	**0.9226**	**0.8453**	**0.8811**	**0.9623**	**0.9164**	**0.8331**	**0.8616**
Imp.	+37.07%	+33.66%	+33.55%	+20.36%	+89.77%	+77.98%	+77.41%	+83.88%	+88.56%	+76.78%	+75.97%	+81.13%

MIRAGE-SIG, Gemini-2.0-flash-lite

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.4149	0.5000	0.0000	0.0081	0.3951	0.5005	0.0232	0.0173	0.3941	0.5000	0.0000	0.0096
LogRank [25]	0.4167	0.5000	0.0000	0.0091	0.3817	0.5000	0.0000	0.0162	0.3848	0.5000	0.0000	0.0096
Entropy [14]	0.7415	0.6819	0.3639	0.2175	0.6534	0.6138	0.2348	0.1694	0.6658	0.6157	0.2535	0.1815
RoBERTa-Base [32] ♠	0.4224	0.5000	0.0000	0.0183	0.4264	0.5000	0.0000	0.0129	0.4604	0.5000	0.0000	0.0372
RoBERTa-Large [32] ♠	0.3609	0.5000	0.0000	0.0142	0.4813	0.5027	0.0162	0.0464	0.5338	0.5234	0.0475	0.0403
LRR [46]	0.4415	0.5000	0.0000	0.0274	0.3475	0.5000	0.0000	0.0205	0.3658	0.5000	0.0000	0.0127
DNA-GPT [58] ◇	0.5780	0.5742	0.1532	0.0417	0.4567	0.5000	0.0000	0.0194	0.4447	0.5000	0.0000	0.0191
NPR [46] ◇	0.6103	0.6184	0.2723	0.0132	0.5199	0.5431	0.1365	0.0205	0.5008	0.5356	0.1117	0.0138
DetectGPT [34] ◇	0.6321	0.6250	0.2832	0.0163	0.5712	0.5766	0.1749	0.0388	0.5473	0.5536	0.1398	0.0234
Fast-DetectGPT [7]	0.8418	0.7581	0.5178	0.4421	0.6724	0.6311	0.2623	0.1780	0.6705	0.6322	0.2753	0.1900
ImBD [9] ♠	0.8564	0.7769	0.5562	0.3872	0.6971	0.6451	0.3156	0.2621	0.6816	0.6391	0.3121	0.2813
DetectAnyLLM(ours) ♠	**0.9358**	**0.8984**	**0.7968**	**0.7104**	**0.9606**	**0.9186**	**0.8381**	**0.8781**	**0.9513**	**0.9002**	**0.8011**	**0.8291**
Imp.	+55.26%	+54.44%	+54.21%	+48.09%	+87.01%	+77.05%	+76.34%	+83.48%	+84.72%	+72.35%	+71.09%	+76.22%

Table 20: Generator: Doubao1.5pro, Grok2. "Imp.": Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$.

MIRAGE-DIG, Doubao1.5pro

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.4877	0.5291	0.0707	0.0040	0.4548	0.5073	0.0258	0.0093	0.4377	0.5000	0.0000	0.0105
LogRank [25]	0.5231	0.5486	0.1033	0.0100	0.4496	0.5027	0.0258	0.0093	0.4395	0.5000	0.0000	0.0105
Entropy [14]	0.6627	0.6249	0.2944	0.1073	0.5673	0.5650	0.1887	0.1273	0.5816	0.5634	0.2068	0.1327
RoBERTa-Base [32] ♠	0.4880	0.5045	0.0391	0.0421	0.4709	0.5000	0.0000	0.0305	0.5464	0.5314	0.0757	0.0780
RoBERTa-Large [32] ♠	0.4792	0.5025	0.0448	0.0371	0.5404	0.5345	0.0690	0.0584	0.6026	0.5669	0.1467	0.1013
LRR [46]	0.6335	0.6103	0.2214	0.0712	0.4315	0.5000	0.0000	0.0146	0.4516	0.5006	0.0241	0.0175
DNA-GPT [58] ♢	0.6432	0.6204	0.2529	0.0662	0.5055	0.5345	0.0707	0.0186	0.4844	0.5087	0.0418	0.0407
NPR [46] ♢	0.5756	0.5832	0.1933	0.0150	0.4475	0.5192	0.0659	0.0265	0.4586	0.5169	0.0615	0.0233
DetectGPT [34] ♢	0.5303	0.5667	0.1746	0.0040	0.4883	0.5305	0.0772	0.0358	0.4810	0.5215	0.0761	0.0244
Fast-DetectGPT [7]	0.8007	0.7282	0.4569	**0.3270**	0.5753	0.5590	0.1312	0.0889	0.5470	0.5518	0.1039	0.0477
ImBD [9] ♠	0.8150	0.7452	0.5195	0.2508	0.6552	0.6194	0.2507	0.1976	0.6091	0.5832	0.1703	0.1048
DetectAnyLLM(ours) ♠	**0.8165**	**0.7603**	**0.5291**	0.3039	**0.8795**	**0.8322**	**0.6683**	**0.6419**	**0.7609**	**0.7183**	**0.4576**	**0.3935**
Imp.	+0.80%	+5.91%	+2.00%	–	+65.06%	+55.92%	+55.74%	+55.37%	+38.84%	+32.40%	+31.62%	+30.07%

MIRAGE-SIG, Doubao1.5pro

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.5010	0.5503	0.1150	0.0050	0.4742	0.5161	0.0346	0.0147	0.4436	0.5000	0.0000	0.0069
LogRank [25]	0.5330	0.5683	0.1491	0.0080	0.4677	0.5027	0.0259	0.0188	0.4442	0.5000	0.0000	0.0069
Entropy [14]	0.6532	0.6271	0.2804	0.0945	0.5647	0.5516	0.1573	0.0992	0.5831	0.5677	0.2145	0.1353
RoBERTa-Base [32] ♠	0.4704	0.5141	0.0462	0.0543	0.4572	0.5000	0.0000	0.0295	0.5370	0.5269	0.0802	0.0814
RoBERTa-Large [32] ♠	0.4842	0.5075	0.0705	0.0452	0.5342	0.5228	0.0850	0.0818	0.5943	0.5722	0.1449	0.0917
LRR [46]	0.6346	0.6151	0.2313	0.0724	0.4482	0.5000	0.0000	0.0228	0.4516	0.5000	0.0000	0.0218
DNA-GPT [58] ♢	0.6585	0.6317	0.2682	0.0834	0.5408	0.5436	0.0871	0.0389	0.5006	0.5126	0.0258	0.0344
NPR [46] ♢	0.5742	0.5940	0.2161	0.0141	0.5011	0.5308	0.0952	0.0241	0.4539	0.5120	0.0386	0.0183
DetectGPT [34] ♢	0.5291	0.5719	0.1949	0.0020	0.5430	0.5483	0.1182	0.0201	0.4826	0.5189	0.0479	0.0229
Fast-DetectGPT [7]	0.7906	0.7211	0.4425	0.3146	0.6173	0.5932	0.1865	0.1193	0.5775	0.5568	0.1270	0.0780
ImBD [9] ♠	0.8055	0.7307	0.4691	0.2503	0.6789	0.6280	0.2652	0.1944	0.6258	0.5981	0.2016	0.1342
DetectAnyLLM(ours) ♠	**0.8594**	**0.8095**	**0.6246**	**0.4643**	**0.8920**	**0.8271**	**0.6561**	**0.6327**	**0.7605**	**0.7144**	**0.4468**	**0.4002**
Imp.	+27.73%	+29.29%	+29.29%	+21.85%	+66.37%	+53.51%	+53.21%	+54.41%	+36.01%	+28.96%	+29.57%	+30.64%

MIRAGE-DIG, Grok2

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.5837	0.5876	0.1872	0.0100	0.4545	0.5021	0.0229	0.0231	0.4441	0.5000	0.0000	0.0190
LogRank [25]	0.5964	0.5906	0.1953	0.0140	0.4417	0.5000	0.0000	0.0200	0.4333	0.5000	0.0000	0.0179
Entropy [14]	0.6154	0.6041	0.2320	0.0701	0.5466	0.5478	0.1168	0.0704	0.5579	0.5580	0.1233	0.0770
RoBERTa-Base [32] ♠	0.5690	0.5711	0.1797	0.1311	0.4921	0.5042	0.0172	0.0494	0.4621	0.5000	0.0000	0.0411
RoBERTa-Large [32] ♠	0.4974	0.5210	0.0549	0.0541	0.5505	0.5394	0.0789	0.0557	0.5568	0.5469	0.0940	0.0570
LRR [46]	0.6278	0.6001	0.2006	0.0841	0.4041	0.5000	0.0000	0.0179	0.4042	0.5000	0.0000	0.0116
DNA-GPT [58] ♢	0.6971	0.6582	0.3215	0.1131	0.4612	0.5026	0.0324	0.0252	0.4681	0.5047	0.0325	0.0243
NPR [46] ♢	0.6492	0.6512	0.3476	0.0150	0.4634	0.5184	0.0943	0.0252	0.4350	0.5084	0.0609	0.0148
DetectGPT [34] ♢	0.6595	0.6507	0.3640	0.0140	0.4927	0.5320	0.1188	0.0252	0.4631	0.5148	0.0755	0.0253
Fast-DetectGPT [7]	0.9074	0.8363	0.6730	0.5976	0.5309	0.5373	0.0746	0.0599	0.5274	0.5295	0.0690	0.0496
ImBD [9] ♠	0.8608	0.7828	0.5777	0.3784	0.6094	**0.5893**	0.1813	0.1544	0.6287	0.5918	0.2250	0.1835
DetectAnyLLM(ours) ♠	**0.9323**	**0.8844**	**0.7690**	0.5856	**0.9152**	**0.8713**	**0.7450**	**0.7405**	**0.9257**	**0.8834**	**0.7692**	**0.7890**
Imp.	+26.87%	+29.36%	+29.36%	–	+78.30%	+68.67%	+68.85%	+69.32%	+79.99%	+71.45%	+70.22%	+74.16%

MIRAGE-SIG, Grok2

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.5995	0.6060	0.2332	0.0180	0.4524	0.5000	0.0000	0.0160	0.4274	0.5000	0.0000	0.0159
LogRank [25]	0.6087	0.6050	0.2369	0.0250	0.4366	0.5000	0.0000	0.0139	0.4181	0.5000	0.0000	0.0180
Entropy [14]	0.6037	0.5975	0.2280	0.0450	0.5521	0.5438	0.1002	0.0544	0.5793	0.5693	0.1466	0.0921
RoBERTa-Base [32] ♠	0.5509	0.5605	0.1683	0.1330	0.4425	0.5000	0.0000	0.0235	0.4993	0.5095	0.0255	0.0434
RoBERTa-Large [32] ♠	0.4871	0.5230	0.0909	0.0850	0.5136	0.5117	0.0548	0.0598	0.5573	0.5487	0.0981	0.0540
LRR [46]	0.6355	0.6045	0.2107	0.0860	0.3927	0.5005	0.0103	0.0107	0.3964	0.5000	0.0000	0.0116
DNA-GPT [58] ♢	0.7035	0.6550	0.3172	0.1310	0.4731	0.5123	0.0453	0.0267	0.4491	0.5026	0.0325	0.0233
NPR [46] ♢	0.6515	0.6555	0.3259	0.0160	0.4655	0.5149	0.0727	0.0288	0.4446	0.5143	0.0758	0.0201
DetectGPT [34] ♢	0.6694	0.6615	0.3520	0.0110	0.4990	0.5299	0.0921	0.0224	0.4736	0.5164	0.0682	0.0275
Fast-DetectGPT [7]	0.8946	0.8125	0.6281	0.5790	0.5304	0.5277	0.0718	0.0555	0.5245	0.5291	0.0703	0.0561
ImBD [9] ♠	0.8614	0.7795	0.5608	0.3910	0.6156	0.5993	0.2047	0.1483	0.6067	0.5862	0.1939	0.1397
DetectAnyLLM(ours) ♠	**0.9532**	**0.9240**	**0.8485**	**0.7830**	**0.9231**	**0.8789**	**0.7615**	**0.7769**	**0.9209**	**0.8688**	**0.7393**	**0.7291**
Imp.	+55.61%	+59.47%	+59.26%	+48.46%	+79.99%	+69.77%	+70.01%	+73.81%	+79.88%	+68.29%	+67.65%	+68.51%

Table 21: Generator: Qwen2.5-7B/R1-Distill. "Imp.": Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$.

MIRAGE-DIG, Qwen2.5-7B-Instruct

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.6303	0.6260	0.2634	0.0260	0.4915	0.5223	0.0460	0.0319	0.4605	0.5026	0.0063	0.0201
LogRank [25]	0.6513	0.6375	0.2755	0.0312	0.4833	0.5176	0.0370	0.0277	0.4537	0.5000	0.0000	0.0180
Entropy [14]	0.6164	0.6078	0.2305	0.0771	0.5493	0.5441	0.1845	0.1138	0.5729	0.5618	0.1744	0.0993
RoBERTa-Base [32] ♠	0.6959	0.6568	0.3370	0.2948	0.5456	0.5324	0.0771	0.0596	0.5699	0.5665	0.1437	0.0813
RoBERTa-Large [32] ♠	0.6896	0.6448	0.2898	0.1969	0.6368	0.6117	0.2237	0.1074	0.6536	0.6008	0.2046	0.1341
LRR [46]	0.7032	0.6615	0.3236	0.1000	0.4498	0.5000	0.0000	0.0245	0.4335	0.5000	0.0000	0.0264
DNA-GPT [58] ♢	0.7173	0.6823	0.3689	0.0969	0.5125	0.5234	0.0512	0.0394	0.4841	0.5079	0.0325	0.0296
NPR [46] ♢	0.7013	0.6776	0.3779	0.0323	0.5427	0.5585	0.1282	0.0426	0.4799	0.5185	0.0617	0.0317
DetectGPT [34] ♢	0.7127	0.6849	0.3883	0.0573	0.5584	0.5665	0.1419	0.0415	0.4920	0.5296	0.1020	0.0306
Fast-DetectGPT [7]	**0.9451**	**0.8786**	**0.7576**	**0.7698**	0.6506	0.6213	0.2453	0.1287	0.6125	0.5834	0.1742	0.1130
ImBD [9] ♠	0.7033	0.6594	0.3544	0.3115	0.6133	0.5904	0.1969	0.1596	0.6114	0.5908	0.2047	0.1626
DetectAnyLLM(ours) ♠	0.9252	0.8688	0.7384	0.6052	**0.8638**	**0.7995**	**0.6143**	**0.6170**	**0.8664**	**0.8173**	**0.6371**	**0.6304**
Imp.	–	–	–	–	+61.03%	+47.05%	+48.90%	+54.43%	+61.43%	+54.23%	+54.38%	+55.86%

MIRAGE-SIG, Qwen2.5-7B-Instruct

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.6404	0.6375	0.2885	0.0223	0.5047	0.5251	0.0519	0.0320	0.4625	0.5037	0.0074	0.0212
LogRank [25]	0.6613	0.6492	0.3006	0.0340	0.4968	0.5208	0.0423	0.0320	0.4582	0.5000	0.0000	0.0201
Entropy [14]	0.6051	0.5977	0.2278	0.0701	0.5499	0.5421	0.1850	0.1109	0.5803	0.5573	0.1951	0.1241
RoBERTa-Base [32] ♠	0.6832	0.6502	0.3243	0.2335	0.5474	0.5341	0.0896	0.0618	0.5849	0.5668	0.1531	0.1092
RoBERTa-Large [32] ♠	0.6801	0.6343	0.2790	0.1964	0.6305	0.5975	0.1968	0.1279	0.6621	0.6262	0.2524	0.1092
LRR [46]	0.7121	0.6699	0.3401	0.1369	0.4622	0.5027	0.0231	0.0309	0.4432	0.5000	0.0000	0.0276
DNA-GPT [58] ♢	0.7349	0.6948	0.3975	0.0913	0.5193	0.5304	0.0674	0.0437	0.4767	0.5011	0.0103	0.0318
NPR [46] ♢	0.7077	0.6948	0.4114	0.0393	0.5498	0.5581	0.1297	0.0458	0.4982	0.5323	0.0908	0.0308
DetectGPT [34] ♢	0.7007	0.6725	0.3729	0.0393	0.5675	0.5661	0.1599	0.0512	0.5124	0.5403	0.1249	0.0233
Fast-DetectGPT [7]	**0.9398**	0.8758	0.7533	**0.7707**	0.6643	0.6253	0.2535	0.1578	0.6395	0.6050	0.2112	0.1379
ImBD [9] ♠	0.6995	0.6529	0.3484	0.3153	0.6135	0.5933	0.2307	0.1887	0.6125	0.5923	0.1964	0.1580
DetectAnyLLM(ours) ♠	0.9387	**0.8997**	**0.7994**	0.7325	**0.8753**	**0.8140**	**0.6309**	**0.6098**	**0.8714**	**0.8150**	**0.6361**	**0.6278**
Imp.	–	+19.23%	+18.68%	–	+62.85%	+50.36%	+50.56%	+51.91%	+61.95%	+50.50%	+51.32%	+55.79%

MIRAGE-DIG, Qwen2.5-7B-Instruct-R1-Distill

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.6056	0.6112	0.2236	0.0110	0.5148	0.5403	0.0861	0.0285	0.5025	0.5158	0.0382	0.0294
LogRank [25]	0.6209	0.6161	0.2354	0.0220	0.5051	0.5285	0.0615	0.0285	0.4965	0.5124	0.0271	0.0238
Entropy [14]	0.6075	0.5972	0.2423	0.0819	0.5483	0.5452	0.1251	0.0855	0.5476	0.5402	0.1042	0.0769
RoBERTa-Base [32] ♠	0.7125	0.6693	0.3600	0.3142	0.5952	0.5713	0.1662	0.1326	0.6098	0.5871	0.1801	0.1041
RoBERTa-Large [32] ♠	0.7827	0.7194	0.4414	0.3839	0.7362	0.6735	0.3474	0.2305	0.7330	0.6708	0.3419	0.2138
LRR [46]	0.6569	0.6247	0.2525	0.0746	0.4698	0.5006	0.0111	0.0273	0.4762	0.5017	0.0238	0.0226
DNA-GPT [58] ♢	0.6482	0.6235	0.2501	0.0905	0.5148	0.5297	0.0879	0.0446	0.4872	0.5107	0.0412	0.0238
NPR [46] ♢	0.6416	0.6479	0.3286	0.0257	0.5137	0.5502	0.1423	0.0335	0.4991	0.5288	0.0870	0.0362
DetectGPT [34] ♢	0.6480	0.6449	0.3276	0.0281	0.5422	0.5582	0.1511	0.0409	0.5157	0.5390	0.0958	0.0385
Fast-DetectGPT [7]	0.9207	0.8454	0.6938	**0.6626**	0.7091	0.6543	0.3110	0.2714	0.6665	0.6295	0.2742	0.1912
ImBD [9] ♠	0.7347	0.6705	0.4022	0.3778	0.6698	0.6388	0.3202	0.2788	0.6339	0.6114	0.2673	0.2319
DetectAnyLLM(ours) ♠	**0.9332**	**0.8778**	**0.7556**	0.6577	**0.8962**	**0.8445**	**0.6912**	**0.6766**	**0.8758**	**0.8167**	**0.6427**	**0.6210**
Imp.	+15.74%	+20.95%	+20.19%	–	+60.66%	+52.37%	+52.69%	+55.15%	+53.48%	+44.33%	+45.72%	+50.66%

MIRAGE-SIG, Qwen2.5-7B-Instruct-R1-Distill

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.6183	0.6190	0.2462	0.0110	0.4994	0.5212	0.0425	0.0218	0.4862	0.5137	0.0280	0.0102
LogRank [25]	0.6348	0.6209	0.2531	0.0208	0.4932	0.5139	0.0282	0.0194	0.4789	0.5097	0.0208	0.0114
Entropy [14]	0.5841	0.5842	0.2179	0.0549	0.5621	0.5484	0.1204	0.0944	0.5596	0.5535	0.1502	0.0774
RoBERTa-Base [32] ♠	0.7115	0.6697	0.3658	0.2821	0.5758	0.5617	0.1558	0.1138	0.5896	0.5700	0.1768	0.1502
RoBERTa-Large [32] ♠	0.7808	0.7131	0.4293	0.3370	0.7082	0.6465	0.3040	0.2288	0.7265	0.6706	0.3430	0.2071
LRR [46]	0.6778	0.6404	0.2852	0.0745	0.4663	0.5012	0.0246	0.0291	0.4610	0.5006	0.0239	0.0159
DNA-GPT [58] ♢	0.6659	0.6374	0.2809	0.0904	0.4948	0.5236	0.0571	0.0266	0.4776	0.5142	0.0579	0.0375
NPR [46] ♢	0.6475	0.6471	0.3331	0.0281	0.5173	0.5381	0.1107	0.0230	0.4915	0.5301	0.0889	0.0250
DetectGPT [34] ♢	0.6615	0.6416	0.3127	0.0379	0.5420	0.5539	0.1267	0.0375	0.5092	0.5296	0.1173	0.0319
Fast-DetectGPT [7]	0.9192	0.8462	0.6923	0.6874	0.7019	0.6429	0.3162	0.2627	0.6624	0.6183	0.2419	0.1593
ImBD [9] ♠	0.7005	0.6709	0.4018	0.3736	0.6638	0.6380	0.2987	0.2542	0.6658	0.6320	0.2840	0.2378
DetectAnyLLM(ours) ♠	**0.9524**	**0.9176**	**0.8353**	**0.8059**	**0.8906**	**0.8305**	**0.6698**	**0.6768**	**0.8673**	**0.8146**	**0.6440**	**0.6246**
Imp.	+41.06%	+46.43%	+46.46%	+37.89%	+62.51%	+52.05%	+51.72%	+56.16%	+51.47%	+43.70%	+45.82%	+50.75%

Table 22: Generator: LlaMa3.1-8B/R1-Distill. "Imp.": Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$.

Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
MIRAGE-DIG, LlaMa3.1-8B-Instruct												
Likelihood [44]	0.7698	0.7395	0.4823	0.1247	0.6213	0.6195	0.2509	0.0260	0.5904	0.5988	0.2174	0.0186
LogRank [25]	0.7944	0.7588	0.5212	0.1865	0.6093	0.6008	0.2217	0.0228	0.5811	0.5837	0.1864	0.0221
Entropy [14]	0.5764	0.5690	0.1623	0.0552	0.5324	0.5545	0.1345	0.0374	0.5726	0.5756	0.1799	0.0616
RoBERTa-Base [32] ♠	0.8327	0.7682	0.5403	0.5121	0.5748	0.5496	0.1175	0.1073	0.6350	0.6099	0.2532	0.2035
RoBERTa-Large [32] ♠	0.7945	0.7246	0.4697	0.4161	0.6461	0.6220	0.2495	0.0943	0.6999	0.6506	0.3040	0.1651
LRR [46]	0.8405	0.7710	0.5436	0.4029	0.5503	0.5512	0.1197	0.0293	0.5408	0.5448	0.0909	0.0384
DNA-GPT [58] ♢	0.8741	0.8151	0.6308	0.4625	0.6908	0.6528	0.3298	0.0992	0.6239	0.6035	0.2148	0.0558
NPR [46] ♢	0.7772	0.7467	0.5155	0.0695	0.5993	0.6065	0.2521	0.0211	0.6029	0.5983	0.2418	0.0244
DetectGPT [34] ♢	0.7984	0.7610	0.5471	0.0673	0.6516	0.6398	0.3069	0.0325	0.6509	0.6320	0.2990	0.0407
Fast-DetectGPT [7]	**0.9944**	**0.9741**	**0.9482**	**0.9845**	0.8546	0.7732	0.5515	0.5057	0.8682	0.7983	0.5965	0.5256
ImBD [9] ♠	0.8708	0.7787	0.5675	0.4691	0.7267	0.6772	0.3798	0.3447	0.7212	0.6715	0.3894	0.3302
DetectAnyLLM(ours) ♠	0.9235	0.8499	0.7002	0.6645	**0.9467**	**0.8919**	**0.7839**	**0.7935**	**0.9678**	**0.9198**	**0.8417**	**0.8872**
Imp.	–	–	–	–	+63.32%	+52.33%	+51.81%	+58.22%	+75.58%	+60.23%	+60.77%	+76.23%
MIRAGE-SIG, LlaMa3.1-8B-Instruct												
Likelihood [44]	0.7701	0.7461	0.4970	0.1057	0.6071	0.6153	0.2497	0.0265	0.5754	0.5935	0.1979	0.0150
LogRank [25]	0.7934	0.7500	0.5073	0.1641	0.5951	0.5995	0.2108	0.0216	0.5683	0.5751	0.1734	0.0162
Entropy [14]	0.5545	0.5600	0.1516	0.0595	0.5632	0.5713	0.1622	0.0348	0.5732	0.5768	0.1772	0.0612
RoBERTa-Base [32] ♠	0.8226	0.7698	0.5612	0.5055	0.5325	0.5390	0.0955	0.0879	0.6481	0.6195	0.2689	0.2182
RoBERTa-Large [32] ♠	0.7850	0.7230	0.4492	0.3678	0.6316	0.6078	0.2188	0.1161	0.7055	0.6542	0.3091	0.1709
LRR [46]	0.8363	0.7671	0.5384	0.4042	0.5321	0.5398	0.0899	0.0265	0.5391	0.5410	0.0983	0.0393
DNA-GPT [58] ♢	0.8866	0.8271	0.6547	0.4901	0.6783	0.6468	0.2936	0.0879	0.6061	0.5901	0.1969	0.0450
NPR [46] ♢	0.7856	0.7561	0.5465	0.0628	0.5999	0.6012	0.2367	0.0282	0.5888	0.5987	0.2250	0.0162
DetectGPT [34] ♢	0.7999	0.7643	0.5360	0.0308	0.6508	0.6260	0.2782	0.0332	0.6450	0.6253	0.2810	0.0300
Fast-DetectGPT [7]	**0.9931**	**0.9725**	**0.9450**	**0.9791**	0.8641	0.7828	0.5706	0.5307	0.8519	0.7760	0.5533	0.4988
ImBD [9] ♠	0.8757	0.7880	0.5798	0.4824	0.7514	0.6915	0.4019	0.3516	0.7152	0.6617	0.3725	0.3372
DetectAnyLLM(ours) ♠	0.9492	0.9091	0.8185	0.7709	**0.9493**	**0.8905**	**0.7825**	**0.8043**	**0.9589**	**0.9042**	**0.8090**	**0.8395**
Imp.	–	–	–	–	+62.71%	+49.62%	+49.34%	+58.30%	+72.26%	+57.22%	+57.24%	+67.97%
MIRAGE-DIG, LlaMa3.1-8B-Instruct-R1-Distill												
Likelihood [44]	0.6583	0.6445	0.3108	0.0095	0.5952	0.5922	0.2220	0.0301	0.5359	0.5509	0.1170	0.0284
LogRank [25]	0.6726	0.6522	0.3143	0.0166	0.5910	0.5869	0.1945	0.0235	0.5282	0.5410	0.0953	0.0252
Entropy [14]	0.5717	0.5904	0.2487	0.0392	0.4731	0.5235	0.0910	0.0366	0.5138	0.5383	0.0984	0.0438
RoBERTa-Base [32] ♠	0.7288	0.6790	0.3734	0.3329	0.5680	0.5660	0.1803	0.1229	0.5836	0.5755	0.1562	0.1072
RoBERTa-Large [32] ♠	0.7414	0.6748	0.3690	0.2723	0.6510	0.6190	0.2380	0.1281	0.6647	0.6214	0.2431	0.1400
LRR [46]	0.7062	0.6522	0.3065	0.1617	0.5596	0.5588	0.1244	0.0222	0.4977	0.5131	0.0354	0.0339
DNA-GPT [58] ♢	0.6855	0.6576	0.3238	0.0892	0.5585	0.5588	0.1312	0.0392	0.5109	0.5246	0.0593	0.0339
NPR [46] ♢	0.6476	0.6468	0.3011	0.0131	0.5575	0.5725	0.1720	0.0484	0.5016	0.5306	0.0968	0.0263
DetectGPT [34] ♢	0.6546	0.6457	0.3247	0.0107	0.5843	0.5889	0.2114	0.0366	0.5276	0.5492	0.1126	0.0252
Fast-DetectGPT [7]	0.9223	0.8484	0.6976	0.6754	0.7160	0.6660	0.3363	0.2209	0.6426	0.6034	0.2176	0.1477
ImBD [9] ♠	0.7661	0.7081	0.4537	0.4328	0.6463	0.6131	0.2858	0.2431	0.6261	0.6034	0.2611	0.2287
DetectAnyLLM(ours) ♠	**0.9541**	**0.8995**	**0.7995**	**0.7800**	**0.8949**	**0.8359**	**0.6721**	**0.6523**	**0.8855**	**0.8233**	**0.6554**	**0.6554**
Imp.	+40.95%	+33.73%	+33.68%	+32.23%	+63.00%	+50.88%	+50.61%	+54.06%	+65.85%	+53.32%	+53.36%	+55.32%
MIRAGE-SIG, LlaMa3.1-8B-Instruct-R1-Distill												
Likelihood [44]	0.6540	0.6486	0.3186	0.0133	0.6019	0.5920	0.2049	0.0363	0.5411	0.5569	0.1167	0.0184
LogRank [25]	0.6707	0.6594	0.3254	0.0242	0.5962	0.5803	0.1821	0.0337	0.5350	0.5450	0.0930	0.0238
Entropy [14]	0.5621	0.5815	0.2278	0.0314	0.4736	0.5207	0.0813	0.0350	0.5050	0.5336	0.1286	0.0390
RoBERTa-Base [32] ♠	0.7119	0.6781	0.3721	0.2717	0.5722	0.5531	0.1383	0.1166	0.5891	0.5720	0.1709	0.1181
RoBERTa-Large [32] ♠	0.7276	0.6649	0.3345	0.2766	0.6661	0.6211	0.2467	0.1710	0.6808	0.6376	0.2755	0.1463
LRR [46]	0.7070	0.6673	0.3491	0.1196	0.5612	0.5544	0.1094	0.0453	0.5108	0.5200	0.0508	0.0368
DNA-GPT [58] ♢	0.6989	0.6606	0.3215	0.0809	0.5693	0.5680	0.1498	0.0544	0.5122	0.5276	0.0907	0.0238
NPR [46] ♢	0.6567	0.6479	0.3291	0.0121	0.5764	0.5771	0.1859	0.0389	0.5224	0.5379	0.1224	0.0368
DetectGPT [34] ♢	0.6618	0.6582	0.3413	0.0085	0.6004	0.5933	0.2188	0.0544	0.5405	0.5504	0.1583	0.0358
Fast-DetectGPT [7]	0.9189	0.8424	0.6850	0.6510	0.7247	0.6645	0.3315	0.2396	0.6474	0.6051	0.2194	0.1603
ImBD [9] ♠	0.7669	0.6987	0.4448	0.4263	0.6606	0.6276	0.3144	0.2617	0.6309	0.6056	0.2633	0.2297
DetectAnyLLM(ours) ♠	**0.9642**	**0.9245**	**0.8497**	**0.8563**	**0.8743**	**0.8180**	**0.6439**	**0.6127**	**0.8776**	**0.8207**	**0.6470**	**0.6186**
Imp.	+55.90%	+52.11%	+52.28%	+58.82%	+54.32%	+45.75%	+46.73%	+47.54%	+61.66%	+50.52%	+51.27%	+50.49%

Table 23: Generator: QwQ-Plus-32B. "Imp.": Improvement over previous SOTA, computed as $(new - old)/(1.0 - old)$.

MIRAGE-DIG, QwQ-Plus-32B												
Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.3957	0.5083	0.0568	0.0010	0.4576	0.5065	0.0255	0.0156	0.3788	0.5000	0.0000	0.0068
LogRank [25]	0.3715	0.5062	0.0525	0.0041	0.4298	0.5006	0.0255	0.0130	0.3553	0.5000	0.0000	0.0068
Entropy [14]	0.6403	0.6165	0.2723	0.0714	0.5070	0.5234	0.0806	0.0519	0.5786	0.5609	0.1837	0.1229
RoBERTa-Base [32] ♠	0.5096	0.5000	0.0000	0.0021	0.4165	0.5000	0.0000	0.0052	0.4212	0.5000	0.0000	0.0080
RoBERTa-Large [32] ♠	0.2154	0.5000	0.0000	0.0010	0.3700	0.5000	0.0000	0.0091	0.4054	0.5000	0.0000	0.0307
LRR [46]	0.3225	0.5000	0.0000	0.0124	0.3549	0.5000	0.0000	0.0143	0.3031	0.5000	0.0000	0.0068
DNA-GPT [58] ◇	0.3538	0.5124	0.0716	0.0031	0.3982	0.5019	0.0255	0.0156	0.3457	0.5011	0.0337	0.0102
NPR [46] ◇	0.6025	0.6304	0.3242	0.0093	0.5336	0.5617	0.1718	0.0325	0.4818	0.5392	0.1312	0.0148
DetectGPT [34] ◇	0.6970	0.6781	0.3832	0.0569	0.5918	0.5903	0.2013	0.0325	0.5411	0.5557	0.1456	0.0353
Fast-DetectGPT [7]	0.6280	0.5958	0.1930	0.1159	0.4422	0.5006	0.0255	0.0377	0.4078	0.5000	0.0000	0.0250
ImBD [9] ♠	0.9103	0.8318	0.6658	0.5663	0.7506	0.6981	0.4292	0.3714	0.7357	0.6894	0.3923	0.3265
DetectAnyLLM(ours) ♠	**0.9589**	**0.9105**	**0.8238**	**0.8023**	**0.9478**	**0.8929**	**0.7860**	**0.8091**	**0.9444**	**0.9135**	**0.8309**	**0.8749**
Imp.	+54.24%	+46.77%	+47.28%	+54.42%	+79.07%	+64.52%	+62.50%	+69.63%	+78.97%	+72.16%	+72.17%	+81.42%

MIRAGE-SIG, QwQ-Plus-32B												
Methods	Generate				Polish				Rewrite			
	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%	AUROC	Accuracy	MCC	TPR@5%
Likelihood [44]	0.4263	0.5098	0.0658	0.0021	0.4292	0.5026	0.0321	0.0166	0.3755	0.5000	0.0000	0.0045
LogRank [25]	0.4024	0.5062	0.0526	0.0010	0.4040	0.5006	0.0253	0.0179	0.3530	0.5000	0.0000	0.0068
Entropy [14]	0.6143	0.5973	0.2357	0.0566	0.5402	0.5320	0.0976	0.0730	0.5835	0.5600	0.1951	0.1459
RoBERTa-Base [32] ♠	0.4835	0.5000	0.0000	0.0000	0.4174	0.5000	0.0000	0.0051	0.4329	0.5000	0.0000	0.0113
RoBERTa-Large [32] ♠	0.2083	0.5000	0.0000	0.0000	0.3718	0.5000	0.0000	0.0371	0.4072	0.5023	0.0476	0.0294
LRR [46]	0.3548	0.5000	0.0000	0.0051	0.3391	0.5000	0.0000	0.0154	0.3005	0.5000	0.0000	0.0057
DNA-GPT [58] ◇	0.3783	0.5113	0.0684	0.0031	0.3868	0.5006	0.0253	0.0115	0.3430	0.5000	0.0000	0.0057
NPR [46] ◇	0.6165	0.6483	0.3357	0.0175	0.5389	0.5621	0.1705	0.0205	0.4724	0.5373	0.1186	0.0158
DetectGPT [34] ◇	0.7068	0.6874	0.4064	0.0319	0.6029	0.6031	0.2159	0.0359	0.5327	0.5486	0.1323	0.0283
Fast-DetectGPT [7]	0.6228	0.5906	0.1863	0.1123	0.4595	0.5000	0.0000	0.0371	0.3927	0.5000	0.0000	0.0170
ImBD [9] ♠	0.9079	0.8357	0.6718	0.5757	0.7660	0.7138	0.4416	0.3880	0.7527	0.7025	0.4140	0.3575
DetectAnyLLM(ours) ♠	**0.9550**	**0.9212**	**0.8429**	**0.8260**	**0.9400**	**0.8899**	**0.7805**	**0.8092**	**0.9256**	**0.8857**	**0.7730**	**0.8077**
Imp.	+51.15%	+52.04%	+52.12%	+58.98%	+74.38%	+61.52%	+60.69%	+68.83%	+69.92%	+61.60%	+61.27%	+70.07%