Imitate Before Detect: Aligning Machine Stylistic Preference for Machine-Revised Text Detection

Jiaqi Chen[1,2*], Xiaoye Zhu[3,4*], Tianyang Liu[5*], Ying Chen[6], Chen Xinhui[7,8], Yiwen Yuan[9], Chak Tou Leong[10], Zuchao Li[7†], Long Tang[1,3], Lei Zhang[5], Chenyu Yan[11], Guanghao Mei[5], Jie Zhang[1†], Lefei Zhang[7†]

[1]Fudan University
[2]Stanford University
[3]South China University of Technology
[4] NUS (Chongqing) Research Institute
[5]University of California, San Diego
[6]University of lllinois at Urbana-Champaign
[7]Wuhan University
[8]Fenz AI
[9]Carnegie Mellon University
[10]The Hong Kong Polytechnic University
[11]Georgia Institute of Technology

Abstract

Large Language Models (LLMs) have revolutionized text generation, making detecting machine-generated text increasingly challenging. Although past methods have achieved good performance on detecting pure machine-generated text, those detectors have poor performance on distinguishing ***machine-revised text*** (rewriting, expansion, and polishing), which can have only minor changes from its original human prompt. As the content of text may originate from human prompts, detecting machine-revised text often involves identifying distinctive machine styles, *e.g.*, worded favored by LLMs. However, existing methods struggle to detect machine-style phrasing hidden within the content contributed by humans. We propose the ***"Imitate Before Detect" (ImBD)*** approach, which first imitates the machine-style token distribution, and then compares the distribution of the text to be tested with the machine-style distribution to determine whether the text has been machine-revised. To this end, we introduce style preference optimization (SPO), which aligns a scoring LLM model to the preference of text styles generated by machines. The aligned scoring model is then used to calculate the style-conditional probability curvature (Style-CPC), quantifying the log probability difference between the original and conditionally sampled texts for effective detection. We conduct extensive comparisons across various scenarios, encompassing text revisions by six LLMs, four distinct text domains, and three machine revision types. Compared to existing state-of-the-art methods, our method yields a 13% increase in AUC for detecting text revised by open-source LLMs, and improves performance by 5% and 19% for detecting GPT-3.5 and GPT-4o revised text, respectively. Notably, our method surpasses the commercially trained GPT-Zero with just $1,000$ samples and five minutes of SPO, demonstrating its efficiency and effectiveness.

[*]These authors contributed equally.
[†]Corresponding author.

Homepage — machine-text-detection.github.io/ImBD
Code — github.com/Jiaqi-Chen-00/ImBD
Extended version — arxiv.org/abs/2412.10432

1 Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in generating text that is difficult to distinguish from human writing (Brown et al. 2020; Chowdhery et al. 2023; Li et al. 2023; Touvron et al. 2023a,b; OpenAI 2022; Achiam et al. 2023; Bi et al. 2024; Lozhkov et al. 2024).

With the widespread application of these models, their misuse in exams, academic papers, publications, and other contexts has led to concerns in areas such as academic integrity, fake news, and online information verification. As a result, determining whether a text is LLM-assisted or entirely human-written has become crucial (Bao et al. 2023).

In practice, the landscape of LLM-assisted writing extends beyond the widely studied pure generation to also include ***machine-revised text***, where LLMs enhance or modify human-written content (Zhang et al. 2024). This shift results in a more nuanced challenge for detection, as the boundaries between human and machine contributions become increasingly intertwined. Figure 1 (upper) provides comparative examples of human-written, machine-generated text, and machine-revised text. This evolution in LLM-assisted writing necessitates a reevaluation of existing detection approaches.

Previous detection methods (Hans et al. 2024; Mitchell et al. 2023; Bao et al. 2023; Su et al. 2023; Yang et al. 2023; Zhu et al. 2023; Wu et al. 2024) for identifying machine-generated text rely on calculating classification metrics based on token probabilities from pre-trained language models. These methods are built on the assumption that machine-generated texts typically exhibit higher log-likelihoods (He

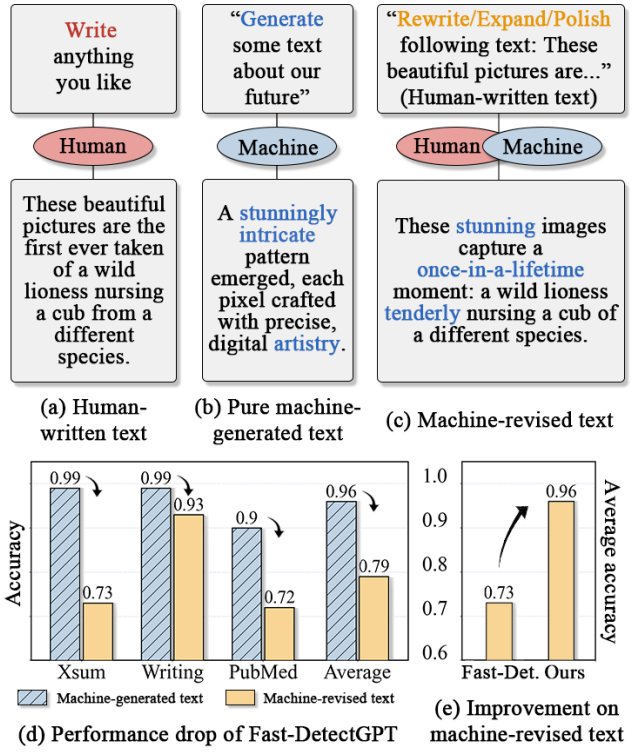

Figure 1: (a-c) Comparative examples of human-written, machine-generated, and machine-revised text. (d) Fast-DetectGPT shows a significant drop in detection accuracy when identifying machine-revised text compared to machine-generated text. (e) Our method brings a noticeable improvement in detecting machine-revised text compared to Fast-DetectGPT. "Fast-Det." denotes "Fast-DetectGPT".

et al. 2024; Holtzman et al. 2020) or negative probability curvatures (Mitchell et al. 2023; Bao et al. 2023) compared to human-written texts. While these approaches effectively capture the characteristics of purely machine-generated text, they struggle to identify machine-revised text that contains human content, such as domain-specific terminology. This is because the human-contributed content can mislead detectors into believing that the text is human-written (Zhang et al. 2024; Sadasivan et al. 2024; He et al. 2024). As a result, these advanced methods experience significant performance drops when detecting machine-revised text (See Figure 1 (d)). We believe that recognizing the distinctive style of machine-revised text, such as machine-preferred filler phrases and rare vocabulary, is key to effectively detecting such texts.

Specifically, the style distinctions between pure-human and machine-revised texts often lie in subtle stylometric features, as demonstrated by examples in Figure 1. Machine revisions exhibit certain characteristic patterns in word choice (*e.g.*, preference for terms like "stunning," "once-in-a-lifetime," and "tenderly"), sentence structures (*e.g.*, more complex subordinate clauses), and organizational methods (*e.g.*, consistent paragraph structuring) (Chawla 2024). However, these style features are difficult to capture and isolate due to the human-contributed content mixed into machine-revised text. Therefore, it is necessary to explicitly model these stylistic features.

Motivated by the challenges and observations above, we propose ***Imitate Before Detect (ImBD)*** which first imitates the style/pattern of machine-revised texts, then measures the distributional differences between the text under evaluation and the machine style, thereby enabling effective detection of machine-revised texts. The ImBD consists of two main steps. First, we introduce ***Style preference optimization (SPO)*** for machine style imitation, which aligns a scoring LLM model to favor the characteristic style of machine-revised text. Specifically, we use pairs of text with identical content – one generated by an LLM and the other written by a human - to adjust the model's token distribution towards a machine-like writing style. Second, we employ the scoring model tuned by step one to calculate the ***Style-conditional probability curvature (Style-CPC)***. This metric quantifies the difference between the log probabilities of the original text and alternative versions produced through conditional probability sampling, enabling effective distinction between human-written and machine-revised content. By combining our style-focused alignment with logit-based detection, our method aims to effectively identify machine-revised text even when dealing with advanced language models like GPT-3.5 or GPT-4o.

We demonstrate the efficiency and effectiveness of our method through extensive comparisons across diverse scenarios. Our results show significant improvements over existing state-of-the-art methods. We achieve an 13% increase in ROAUC for detection on open-source models; 5% and 19% respective increases on GPT-3.5 and GPT-4o, with limited computational resources – just $1,000$ samples and five minutes of SPO training – our approach outperforms the commercially trained GPT-Zero detector.

Our contributions are three-fold:

- We propose the ***Imitate Before Detect*** which first imitates the stylistic preferences of LLMs, then measures the distribution distance to recognize machine-revised text that includes human content.
- We introduce a comprehensive dataset for machine-revised text detection, enabling robust evaluation of detection methods across diverse domains, revision types, and a wide range of mainstream LLMs.
- Our approach achieves 15.16%, 19.68%, and 12.90% higher ROCAUC than the previous state-of-the-art, Fast-DetectGPT, in detecting revised text from GPT-3.5, GPT-4o, and mainstream open-source LLMs respectively with the same inference speed.

2 Method

We elaborate on the methods for addressing the challenge of machine-revised text detection, aiming to differentiate between pure human texts and machine-revised texts.

2.1 Problem Formulation

Let x denote the given text under detection, represented as a sequence of tokens $\{x_i\}_{i=1}^{n}$, where n is the length of the sequence. This text x may either be revised by machine or

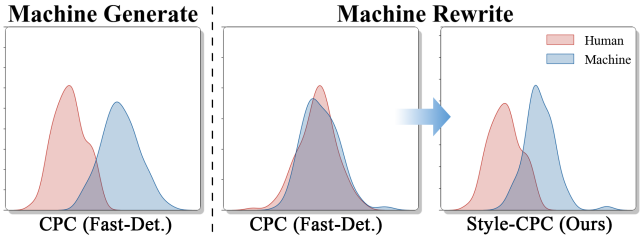

Figure 2: **Impact of *Style-conditional probability curvatures (Style-CPC).*** (Left) Conditional probability curvatures (CPC) from Fast-DetectGPT (denoted as "Fast-Det.") applied to purely machine-generated text; (Middle) Conditional probability curvatures applied to purely machine-revised text; (Right) Style-conditional probability curvatures from ours applied to machine-revised text. The greater the separation between human-written texts (red) and machine-revised texts (blue), the more effective the detection.

authored by a human. Our primary objective is to utilize a scoring model p_θ, which is an autoregressive language model, to ascertain whether the text x is machine-revised (x_m) or human-written (x_h), thereby formulating this problem as a binary classification task. Formally, we aim to construct a decision function $f : x \rightarrow 0, 1$, where the output 0 indicates that the text is human-authored, and 1 signifies that the text is machine-revised.

2.2 Preliminary

Foundation The foundation of machine-generated text detection methods often lies in analyzing the probability distribution of tokens within a given text. This is rooted in the fact that common decoding strategies, such as top-k, top-p, and beam search, favor high-likelihood next tokens in autoregressive generation, while high-quality human language does not necessarily follow high-probability next words (Holtzman et al. 2020).

To quantify the differences between machine-generated text x_m and human-written text x_h, one effective strategy is to measure the discrepancy (δ) between the log probability of the original text and its alternative versions under perturbation (Mitchell et al. 2023) or after resampling (Bao et al. 2023). Let ϕ denote a transformation function that produces an altered version $\tilde{x}$ from the original text x, *i.e.*, $\tilde{x} \sim \phi(x)$. In machine-generated texts, the original tokens often have higher probabilities, and after applying ϕ for token replacement, the probabilities of the new tokens tend to be lower on average. Conversely, human-written texts typically exhibit a more diverse range of token probabilities, leading to a smaller discrepancy after alterations. As a result, this discrepancy tends to be larger for machine-generated text compared to human-written text. Formally, we can express this inequality as:

$$\underbrace{\log p(x_m) - \mathbb{E}_{\tilde{x}_m \sim \phi(x_m)} \log p(\tilde{x}_m)}_{\text{discrepancy of machine-generated text } (\delta_m)} > \underbrace{\log p(x_h) - \mathbb{E}_{\tilde{x}_h \sim \phi(x_h)} \log p(\tilde{x}_h)}_{\text{discrepancy of human-written text } (\delta_h)},$$

where p represents the probability distribution of the source model. The source model can be effectively replaced by a substitute scoring model p_θ in black-box scenarios (Mitchell et al. 2023). This inequality forms the basis for distinguishing between machine-generated and human-written content. Recent studies have demonstrated the effectiveness of this approach in detecting machine-generated text (Mitchell et al. 2023; Bao et al. 2023). In scenarios where the distributions of these discrepancies show a small overlap between machine-generated and human-written texts, this approach can effectively distinguish between the two types of content. As shown in Figure 2 (left), the distribution of the discrepancy for machine-generated text is generally larger than that for human-written text, creating a gap that allows differentiation between the two.

Problem Analysis While the aforementioned approach can be effective for detecting pure machine-generated text, it encounters significant challenges when applied to more nuanced scenarios, particularly in the detection of machine-revised texts. In tasks, such as `rewrite` or `polish`, where machines make small changes on top of human writing, we observe a substantial overlap in the probability distributions of machine-revised and human-written texts, as shown in Figure 2 (right).

This overlap severely compromises the effectiveness of detection methods that rely on the hypothesis. The limitations arise from two key factors. First, when users provide part of the content, the resulting text is not entirely "machine-generated", making probability-based distinctions less effective. Second, advanced LLMs may develop unique stylistic patterns that are not captured by traditional methods. For instance, models like GPT-4 might favor words such as *commendable*, "*embark*", "*delve into*", "*intricate*", *etc.* (Liang et al. 2024; Gray 2024; Chawla 2024), in contexts where a scoring model trained on a general corpus would consider them unexpected. This discrepancy skews the calculation of the probability curvature, leading to values that significantly overlap between machine-revised and human-written texts, making reliable distinction challenging.

These challenges underscore the need for a more nuanced approach to detection that focuses on capturing the subtle stylistic differences between human-written and machine-revised text. Therefore, we propose to learn the characteristic style of machine-revised text by imitating the token distribution output by LLMs. By focusing on style rather than content, we aim to enhance the detector's ability to distinguish between human-written and machine-revised text.

2.3 Imitating via Preference Optimization

Based on the challenges identified in detecting machine-revised text, we observed that the key to effective detection

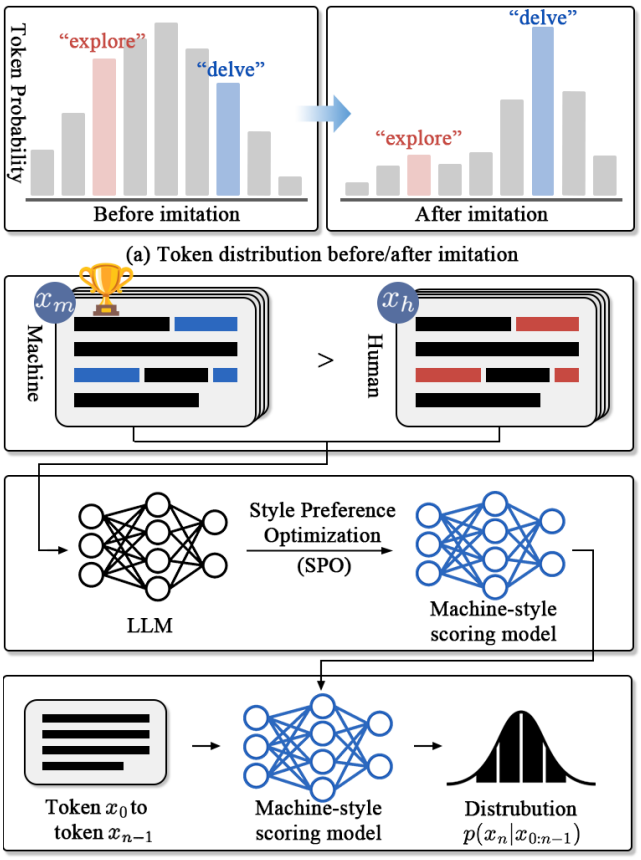

Figure 3: **Imitating the stylistic preferences of LLMs.** (a) Token distribution before and after machine-style imitation, demonstrating a deliberate fine-tuning of the scoring model to bias its token distribution towards a machine writing style (*e.g.*, shifting preferences from common words like "explore" to machine-favored tokens such as "delve"). (b) The pipeline of Style Preference Optimization is applied to align the base scoring model with the style of machine-revised content using paired human-machine texts. This results in a machine-style scoring model, which generates token distributions $p(x_n|x_{0:n-1})$ for each position n, subsequently used for style-conditional probability curvature calculations.

lies in increasing the discrepancy between the probability distributions of machine-revised and human-written texts. To address this, we aim to increase the difference between the discrepancies δ_m and δ_h, as defined earlier. Specifically, our objective is to optimize the scoring model p_θ to better imitate the token distribution with machine style, such that:

$$\max_{p_\theta} \mathbb{E}_{x_m,x_h}[\delta_m - \delta_h].$$

This objective seeks to widen the gap between the discrepancies between machine-revised and human-written texts, making them more distinguishable. To achieve this, we propose a method called style preference optimization, which leverages preference learning to tune the scoring model p_θ towards favoring machine-revised text patterns.

As shown in Figure 3 (b), the core of this method involves constructing preference relations between pairs of texts with equivalent content: one human-written (x_h) and one machine-revised (x_m). These pairs are created through a rewriting process, ensuring that the content remains consistent while the writing style varies. This pairing strategy allows us to isolate and focus on stylistic differences, controlling for content variability. By optimizing the scoring model p_θ to exhibit a stronger preference for the stylistic features of machine-revised text x_m over those of human-written text x_h, we denote this preference as $x_m \succ x_h$. We formulate this preference learning through the lens of reward learning. Assuming an optimal reward function r, we express the preference distribution p^* using the Bradley-Terry model:

$$p^*(x_m \succ x_h) = \sigma(r(x_m) - r(x_h)),$$

where σ is the sigmoid function. This formulation indicates that the probability of preferring machine-revised text over human-written text increases as the reward difference $r(x_m) - r(x_h)$ grows. Following the Direct Preference Optimization (DPO) approach, we reparameterize the reward function r using a closed-form expression based on the optimal policy:

$$r(x) = \beta \log \frac{p_\theta(x)}{p_{\theta_{\text{ref}}(x)}}.$$

Here, $p_{\theta_{\text{ref}}}$ represents a reference model, typically the initial state of p_θ before optimization. By incorporating this reward formulation, we express the probability of preference data with the policy model rather than the reward model. Given a training dataset $\mathcal{D}$ of content-equivalent (x_m, x_h) pairs, we optimize the following objective:

$$\max_{p_\theta} \mathop{\mathbb{E}}_{(x_m,x_h)\sim\mathcal{D}} \left[\log \sigma\left(r(x_m) - r(x_h)\right)\right].$$

By optimizing this objective function, we can adjust the model p_θ to favor the stylistic features of machine-revised texts. This adjustment makes the model more sensitive to the stylistic characteristics of machine-revised text. We denote the optimized model as $\hat{p_\theta}$, representing a machine-style scoring model that is strongly aligned with machine styles.

2.4 Detection via Style Probability Curvature

After aligning our model with machine-revised text styles, we proceed with the detection step using conditional probability curvature (Bao et al. 2023). Specifically, given the machine-style scoring model $\hat{p_\theta}$ and a sampling model q_ϕ, we define the style-conditional probability as:

$$p(\tilde{x}|x) = \prod_j \hat{p_\theta}(\tilde{x}_j|x_{<j}).$$

Here, $\tilde{x}$ is generated by sampling each token x_i from $\hat{p_\theta}(x_i \mid x_{<i})$ without conditioning on other sampled tokens. The *style-conditional probability curvature (Style-CPC)* is quantified as:

$$\mathbf{d}(x, \hat{p_\theta}, q_\phi) = \frac{\log \hat{p_\theta}(x|x) - \tilde{\mu}}{\tilde{\sigma}},$$

where

$$\tilde{\mu} = \mathbb{E}_{\tilde{x}\sim q_\phi(\tilde{x}|x)} \left(\log p_\theta(\tilde{x}_i \mid x)\right) \quad ,$$

$$\tilde{\sigma}^2 = \mathbb{E}_{\tilde{x}\sim q_\phi(\tilde{x}|x)} \left(\log p_\theta(\tilde{x}_i \mid x) - \tilde{\mu}^2\right).$$

This metric $\mathbf{d}(x, \hat{p_\theta}, q_\phi)$ allows us to quantify the log probability difference between the original and alternative sampled texts. Figure 2 illustrates the distribution of $\mathbf{d}$ before and after applying Style-CPC. We observe that using the aligned model to calculate $\mathbf{d}$ significantly reduces the overlap between distributions of human-written and machine-revised texts. This reduced overlap enables us to identify an effective threshold value ϵ, leading to a straightforward classification strategy:

$$f(x) = \begin{cases} 1 & \text{if } \mathbf{d}(x, \hat{p_\theta}, q_\phi) > \epsilon \\ 0 & \text{otherwise} \end{cases},$$

where $f(x) = 1$ indicates machine-revised text, and $f(x) = 0$ signifies human-written text. By combining machine style alignment with probability curvature detection, our method aims to enhance the model's sensitivity to the unique stylistic features of machine-revised texts. Essentially, we tune the scoring model to be more biased towards machine-revised styles, making it 'aware' of the subtle differences between machine and human writing styles. This increased sensitivity allows for a more pronounced separation in the probability curvature distributions of machine and human-authored texts. Consequently, the previously overlapping distributions become more distinct, enabling effective logits-based detection that was previously challenging. This approach shifts the focus from content to style, seeking to address the limitations of traditional methods in detecting outputs from advanced language models and in scenarios with user-provided content.

3 Experiment

3.1 Machine Revision Dataset

Data sources The human-written texts included in the training dataset were crawled from the internet before 2019. The texts are then polished by GPT-3.5.[1] We use 500 pairs of samples for training. The composition of the dataset is 57.3% papers, 14.2% blogs, 4.0% letters and emails, and 2.1% homework. For the test data, we follow Bao et al. (2023); Mitchell et al. (2023), use paragraphs from diverse domains as human-written texts, including *XSum* (Narayan, Cohen, and Lapata 2018) for news articles, *SQuAD* (Fan, Lewis, and Dauphin 2018) for Wikipedia contexts, *WritingPrompts* (Fan, Lewis, and Dauphin 2018) (Abbreviated as "Writing") for story writing, and *PubMedQA* (Jin et al. 2019) for biomedical research question answering. Then, we use the pipeline detailed in the following paragraph to generate correspondent machine-revised text.

Dataset process We design a cohesive two-stage pipeline to revise human-written text.

- **Revision instruction generation**: For each task, instructions are constructed with varying tones and lengths using GPT-3.5. The tone is randomly selected from a set of 10 predefined options, while the instruction length is chosen from the set of $\{15, 30, 50\}$ words. The intuition behind choosing different tones and lengths is to simulate different human behaviors.
- **Paragraph revision**: The generated instruction and the human-written text are then prompted into the LLM to produce the final machine-revised text.

Target LLMs for revision We experiment with four open-source models: Qwen2-7B (Yang et al. 2024), Llama-3-8B (Meta AI 2024), Mixtral-7B (Jiang et al. 2024), and Deepseek-7B (Bi et al. 2024), as well as two proprietary models, GPT-3.5 (OpenAI 2022) and GPT-4o (Achiam et al. 2023). Our choice covers a broad spectrum of user preferences.

Machine revision tasks We evaluate the performance of the detector on three tasks: `rewrite`, `expand`, and `polish`. (i) **`Rewrite`**: The LLM is asked to rewrite the given text while preserving all details. (ii) **`Expand`**: The LLM is asked to expand the original text given a style parameter randomly chosen from a set of 10 options such as formal, literary, *etc*. (iii) **`Polish`**: The LLM is asked to polish/adjust the text based on a randomly picked style. Furthermore, we test our method on the `generate` task used in the common evaluation of machine-generated text detectors, which does not fall under the category of machine-revised text detection. To produce machine-generated text for `generate` task, the LLM is prompted with the first 30 tokens of the human written text, following the design in DetectGPT (Mitchell et al. 2023) and Fast-DetectGPT (Bao et al. 2023).

3.2 Baselines

We compare our method with two lines of method: training-based models, and logit-based models. Following Bao et al. (2023), we use AUROC as a metric to evaluate detection accuracy. (i) **Training-based models** include RoBERTa-base (Liu 2019) and RoBERTa-large (Liu 2019), which is trained on substantial datasets up to 160GB of text data, as well as the commercial detector GPTZero (Tian and Cui 2023), which is trained on massive datasets. (ii) **Logit-based models** include Likelihood (Ippolito et al. 2020) (mean log probabilities), LogRank (Solaiman et al. 2019) (average log of ranks in descending order by probabilities), Entropy (Gehrmann, Strobelt, and Rush 2019) (mean token entropy of the predictive distribution), LRR (Su et al. 2023) (an amalgamation of log probability and log-rank), NPR (Su et al. 2023) (normalized perturbed log-Rank) and DNA-GPT (Yang et al. 2023) (divergent N-Gram Analysis), DetectGPT (Mitchell et al. 2023), and its advanced variant, Fast-DetectGPT (Bao et al. 2023).

Note that Fast-DetectGPT (Bao et al. 2023), the current state-of-the-art approach, also serves as a baseline method that does not involve machine-style imitation.

3.3 Main Results

Detection performance for GPT series We evaluate our method using passages polished by GPT-3.5 and GPT-4o across different domains. As shown in Table 1, our method outperforms Fast-DetectGPT by 15.16% and 19.68% in detecting GPT-3.5 and GPT-4 outputs, respectively, on the

[1]All mentions of GPT-3.5 and GPT-4o in this paper refer to `gpt-3.5-turbo-0125` and `gpt-4o-2024-05-13`, respectively.

Method	Time cost (s/1k words)	GPT-3.5 XSum	GPT-3.5 Writing	GPT-3.5 PubMed	Avg.	GPT-4o XSum	GPT-4o Writing	GPT-4o PubMed	Avg.
RoBERTa-base	**0.07**	0.5806	0.7225	0.4370	0.5800	0.4921	0.4774	0.2496	0.4064
RoBERTa-large	0.11	0.6391	0.7236	0.4848	0.6158	0.4782	0.4708	0.3089	0.4193
Likelihood	0.38	0.4982	0.8788	0.5528	0.6433	0.4396	0.8077	0.4596	0.5690
Entropy	0.35	0.6742	0.3021	0.5662	0.5142	0.6122	0.2802	0.5899	0.4941
LogRank	0.36	0.4711	0.8496	0.5597	0.6268	0.4002	0.7694	0.4472	0.5389
LRR	0.41	0.4016	0.7203	0.5629	0.5616	0.3095	0.6214	0.4710	0.4673
DNA-GPT♢	35.92	0.5338	0.8439	0.3333	0.5703	0.4974	0.7478	0.3151	0.5201
NPR♢	111.99	0.5659	0.8786	0.4246	0.6230	0.5065	0.8444	0.3740	0.5750
DetectGPT♢	111.33	0.6343	0.8793	0.5608	0.6915	0.6217	0.8771	0.5612	0.6867
Fast-Detect-GPT	0.72	0.7312	0.9304	0.7182	0.7933	0.6293	0.8324	0.6175	0.6931
ImBD (Ours)	0.72	**0.9849**	**0.9871**	**0.8626**	**0.9449**	**0.9486**	**0.9468**	**0.7743**	**0.8899**

Table 1: **Detection of *GPT-3.5* and *GPT-4o* polished text**. Typically, the Neo-2.7B (Black et al. 2021) is used as the source for the scoring model. NPR and DetectGPT, on the other hand, utilize T5-3B (Chen et al. 2019) for generating perturbations, whereas Fast-DetectGPT employs GPT-J (Wang and Komatsuzaki 2021) as a surrogate model to generate samples. The ♢ symbol denotes methods that require multiple model invocations, leading to a substantial increase in computational load. Metric: AUROC.

Method	XSum	Writing	PubMed	Avg.
GPTZero	0.9542	0.9711	**0.8800**	0.9351
ImBD (Ours)	**0.9849**	**0.9871**	0.8626	**0.9449**

Table 2: **Compared with GPTZero on detecting *GPT-3.5* polished text.** Metric: AUROC.

Method	Qwen2	Llama-3	Mixtral	Deepseek	Avg.
Likelihood	0.4121	0.6861	0.5881	0.6887	0.5938
Entropy	0.6819	0.5546	0.5741	0.4923	0.5757
LogRank	0.3778	0.6581	0.5498	0.6710	0.5642
LRR	0.3025	0.5519	0.4299	0.6010	0.4713
DNA-GPT	0.5021	0.6809	0.6091	0.7031	0.6238
NPR	0.5388	0.7186	0.5988	0.6551	0.6278
DetectGPT	0.6193	0.7706	0.6826	0.7160	0.6971
Fast-DetectGPT	0.7323	0.8870	0.8164	0.8687	0.8261
ImBD (Ours)	**0.9367**	**0.9767**	**0.9492**	**0.9574**	**0.9550**

Table 3: **Detection on *open-source model* polished text.** AUROC scores are averaged across the XSum, SQuAD, and WritingPrompts datasets. Among them, Qwen2, Mixtral, and Deepseek are 7B models, while Llama-3 is an 8B model.

`polish` task. Furthermore, compared to the supervised detectors RoBERTa-large, our method shows an improvement of 32.91%/47.06% on detecting GPT-3.5 and GPT-4, respectively. Additionally, as shown in Table 2, our method surpasses GPTZero by 0.98%. This indicates that our method is highly efficient in training, achieving superior performance with a small amount of data compared to models trained on much larger datasets. To demonstrate task generalization, we compared performance on the `rewrite` task, where our method outperformed Fast-DetectGPT by 36.96% and 24.29% in detecting GPT-3.5 and GPT-4o outputs, respectively.

Method	Tasks Rewrite	Expand	Polish	Generate	Avg.
Likelihood	0.4073	0.4564	0.6039	0.8939	0.5904
Entropy	0.5840	0.6629	0.5431	0.4129	0.5507
LogRank	0.3868	0.4273	0.5864	0.8925	0.5732
LRR	0.3488	0.3581	0.5183	0.8541	0.5198
DNA-GPT	0.4101	0.4901	0.5847	0.8931	0.5945
NPR	0.3606	0.5139	0.5673	0.8541	0.5740
DetectGPT	0.4060	0.6000	0.6615	0.8985	0.6415
Fast-DetectGPT	0.4499	0.7159	0.7989	0.9706	0.7338
ImBD (Ours)	**0.8739**	**0.9758**	**0.9707**	**0.9996**	**0.9550**

Table 4: **Performance on diverse tasks.** We evaluated the detection performance, measured by average AUROC, of text revised by leading LLMs (Qwen2-7B, Llama-3-8B, Mixtral-7B, Deepseek-7B, GPT-3.5, and GPT-4o) on the XSum dataset.

Detection performance on open-source models The performance on `polish` task by open-source models is shown in Table 3. ImBD achieves the highest average AUROC, outperforming DetectGPT by 25.79%.

Robustness in machine revision and generation As shown in Table 4, our method outperforms the state-of-art Fast-DetectGPT by 22.12% on average across all four tasks. The results showcase the robustness of our approach across various tasks and user instructions.

Inference time and training efficiency Our model is trained for 2 epochs with a learning rate set to 0.0001 and β set to 0.05. Each epoch requires approximately 110 seconds on an L20 (48G) GPU, leading to a total training time of 220.57 seconds. As shown in Table 1 , our method achieves a competitive inference time of 0.72 seconds per 1000 words, matching that of Fast-DetectGPT (154.62× speed-up compared to DetectGPT), but with better performance.

Strategy	GPT-3.5 XSum	Writ.	Pub.	Avg.	GPT-4o XSum	Writ.	Pub.	Avg.
w/o imitate	0.73	0.93	0.72	0.79	0.63	0.83	0.62	0.69
SFT	0.56	0.70	0.70	0.65	0.60	0.74	0.66	0.67
SFT*	0.59	0.70	0.66	0.65	0.61	0.73	0.60	0.65
RLHF	0.70	0.92	0.78	0.80	0.54	0.81	0.64	0.66
ORPO	0.79	0.97	0.81	0.86	0.60	0.87	0.66	0.71
ImBD (Ours)	**0.99**	**0.99**	**0.86**	**0.95**	**0.95**	**0.95**	**0.77**	**0.89**

Table 5: **Ablation on preference optimization.** Comparative performance of SPO, supervised fine-tuning (SFT), RLHF, and ORPO strategies across datasets. Training dataset size: 1,000 samples. "*" denotes trained on 3x samples. "Pub." denotes "PubMed" and "Writ." denotes "WritingPrompts". Metric: AUROC. Task: `Polish`.

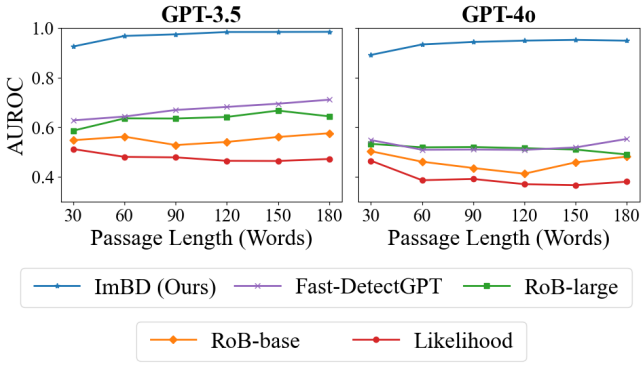

Figure 4: Evaluations of detection accuracy for XSum polished texts trimmed to the specified word count.

3.4 Ablation Study

Ablation on machine-style imitation As shown in Table 5, using fast-DetectGPT as the baseline without imitation, our method improves detection accuracy by 16% and 20% on GPT-3.5 and GPT-4o machine-revised texts, respectively.

Ablation on preference optimization To demonstrate the difference between different optimization methods on ImBD, we compare the performance of SPO against other alignment approaches on `polish` task. As shown in Table 5, ImBD outperformed the SFT variant by 30% on GPT-3.5 and 24% on GPT-4o, even when the SFT variant uses 3x training data. Additionally, ImBD exceeds RLHF and ORPO significantly.

Ablation on text length As shown in Figure 4, our method demonstrates strong performance across passages of varying lengths compared to other methods, with accuracy improving as passage length increases.

4 Related Work

4.1 Machine-Generated Text Detection

Datasets Researchers developed various evaluation benchmarks for machine-generated text detection. Bao et al. (2023) and Mitchell et al. (2023) used the initial 30 tokens from human-written texts across different domains as prompts to generate pure machine-generated text via LLMs. Following this approach, Guo et al. (2023) employed QA datasets as human samples and generated pure machine-generated text using ChatGPT. Building upon the QA framework, researchers (Mitchell et al. 2023; Su et al. 2023; Hu, Chen, and Ho 2023; He et al. 2024; Wang et al. 2024) collected texts generated by LLMs. Verma et al. (2023) focused on creative writing tasks, providing only writing prompts or headlines to generate text with LLMs. However, a significant portion of contemporary machine-generated content involves human input (Zhang et al. 2024). In contrast, our study focuses on the reverse: human-written text revised by LLMs. This practice, where people use AI to enhance, edit, or expand their writing, is increasingly common and accepted in various contexts but remains largely prohibited in academic settings.

Methods While training-based methods (Guo et al. 2023; Chen et al. 2023; Hu, Chen, and Ho 2023) achieved excellent performance due to large-scale data and high-cost training, they tended to overfit and were less effective in detecting the machine-revised text. Existing logit-based approaches (Solaiman et al. 2019; Gehrmann, Strobelt, and Rush 2019; Mitchell et al. 2023) relied on statistical analysis to evaluate information beyond the token level. GLTR (Gehrmann, Strobelt, and Rush 2019) combined a set of metric-based methods to assist human identification. DetectGPT (Mitchell et al. 2023) built on the observation that machine-generated texts occupy regions with steep negative log probability curvature, using this probability curvature to detect whether text originates from LLMs. This concept was further developed and improved in subsequent studies (Su et al. 2023; Mireshghallah et al. 2024; Bao et al. 2023; Zeng et al. 2024). While previous approaches generally relied on overall text features, we propose isolating stylistic features as the basis.

4.2 Preference Optimization

Direct Preference Optimization (Rafailov et al. 2024) can efficiently learn and align preferences from a pair of sampled texts. Yuan et al. (2024); Ethayarajh et al. (2024); Hong, Lee, and Thorne (2024); Park et al. (2024) is primarily for text-generation tasks. However, our study is the first to apply preference optimization to align with a distinct AI style (rather than human preferences) and to use this approach in the context of machine-revised text detection.

5 Conclusion

In this work, we have presented the "***Imitate Before Detect***" paradigm to detect machine-revised text by learning to imitate the writing style of LLMs. Specifically, we have proposed style preference optimization for aligning the detector with machine writing styles and leveraged style-conditional probability curvature to quantify log probability differences for effective detection. We have conducted extensive evaluations, demonstrating significant improvements in detection accuracy compared to existing state-of-the-art methods.

Acknowledgements

We express our gratitude to Zhenyu Ding, Yuanhe Chang, and Longzhi Bing from MercallureAI, Fulong Yang, Yue Wang,

and Yifei Ke for their great support.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint*.

Bao, G.; Zhao, Y.; Teng, Z.; Yang, L.; and Zhang, Y. 2023. Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature. *arXiv preprint*.

Bi, X.; Chen, D.; Chen, G.; Chen, S.; Dai, D.; Deng, C.; Ding, H.; Dong, K.; Du, Q.; Fu, Z.; et al. 2024. Deepseek llm: Scaling open-source language models with longtermism. *arXiv preprint*.

Black, S.; Gao, L.; Wang, P.; Leahy, C.; and Biderman, S. 2021. Gpt-neo: Large scale autoregressive language modeling with mesh-tensorflow. *Computer Science, Linguistics*.

Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J. D.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *NeurIPS*.

Chawla, D. S. 2024. Is ChatGPT corrupting peer review? Telltale words hint at AI use. *Nature*.

Chen, W.; Chen, J.; Qin, P.; Yan, X.; and Wang, W. Y. 2019. Semantically conditioned dialog response generation via hierarchical disentangled self-attention. *arXiv preprint*.

Chen, Y.; Kang, H.; Zhai, V.; Li, L.; Singh, R.; and Raj, B. 2023. Gpt-sentinel: Distinguishing human and chatgpt generated content. *arXiv preprint*.

Chowdhery, A.; Narang, S.; Devlin, J.; Bosma, M.; Mishra, G.; Roberts, A.; Barham, P.; Chung, H. W.; Sutton, C.; Gehrmann, S.; et al. 2023. Palm: Scaling language modeling with pathways. *JMLR*.

Ethayarajh, K.; Xu, W.; Muennighoff, N.; Jurafsky, D.; and Kiela, D. 2024. KTO: Model Alignment as Prospect Theoretic Optimization.

Fan, A.; Lewis, M.; and Dauphin, Y. 2018. Hierarchical Neural Story Generation. In *Association for Computational Linguistics*.

Gehrmann, S.; Strobelt, H.; and Rush, A. M. 2019. Gltr: Statistical detection and visualization of generated text. *arXiv preprint*.

Gray, A. 2024. ChatGPT "contamination": estimating the prevalence of LLMs in the scholarly literature.

Guo, B.; Zhang, X.; Wang, Z.; Jiang, M.; Nie, J.; Ding, Y.; Yue, J.; and Wu, Y. 2023. How Close is ChatGPT to Human Experts? Comparison Corpus, Evaluation, and Detection.

Hans, A.; Schwarzschild, A.; Cherepanova, V.; Kazemi, H.; Saha, A.; Goldblum, M.; Geiping, J.; and Goldstein, T. 2024. Spotting LLMs With Binoculars: Zero-Shot Detection of Machine-Generated Text.

He, X.; Shen, X.; Chen, Z.; Backes, M.; and Zhang, Y. 2024. MGTBench: Benchmarking Machine-Generated Text Detection. *arXiv preprint*.

Holtzman, A.; Buys, J.; Du, L.; Forbes, M.; and Choi, Y. 2020. The Curious Case of Neural Text Degeneration.

Hong, J.; Lee, N.; and Thorne, J. 2024. Orpo: Monolithic preference optimization without reference model. *arXiv preprint*.

Hu, X.; Chen, P.-Y.; and Ho, T.-Y. 2023. RADAR: Robust AI-Text Detection via Adversarial Learning. In *NeurIPS*.

Ippolito, D.; Duckworth, D.; Callison-Burch, C.; and Eck, D. 2020. Automatic Detection of Generated Text is Easiest when Humans are Fooled. In *Association for Computational Linguistics*.

Jiang, A. Q.; Sablayrolles, A.; Roux, A.; Mensch, A.; Savary, B.; Bamford, C.; Chaplot, D. S.; Casas, D. d. l.; Hanna, E. B.; Bressand, F.; et al. 2024. Mixtral of experts. *arXiv preprint*.

Jin, Q.; Dhingra, B.; Liu, Z.; Cohen, W. W.; and Lu, X. 2019. PubMedQA: A Dataset for Biomedical Research Question Answering. In *Conference on Empirical Methods in Natural Language Processing and International Joint Conference on Natural Language Processing*.

Li, R.; Allal, L. B.; Zi, Y.; Muennighoff, N.; Kocetkov, D.; Mou, C.; Marone, M.; Akiki, C.; Li, J.; Chim, J.; et al. 2023. StarCoder: may the source be with you!

Liang, W.; Izzo, Z.; Zhang, Y.; Lepp, H.; Cao, H.; Zhao, X.; Chen, L.; Ye, H.; Liu, S.; Huang, Z.; et al. 2024. Monitoring AI-Modified Content at Scale: A Case Study on the Impact of ChatGPT on AI Conference Peer Reviews.

Liu, Y. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint*.

Lozhkov, A.; Li, R.; Allal, L. B.; Cassano, F.; Lamy-Poirier, J.; Tazi, N.; Tang, A.; Pykhtar, D.; Liu, J.; Wei, Y.; et al. 2024. StarCoder 2 and The Stack v2: The Next Generation.

Meta AI. 2024. Introducing Meta Llama 3: The most capable openly available LLM to date.

Mireshghallah, N.; Mattern, J.; Gao, S.; Shokri, R.; and Berg-Kirkpatrick, T. 2024. Smaller Language Models are Better Zero-shot Machine-Generated Text Detectors. In *the European Chapter of the Association for Computational Linguistics*.

Mitchell, E.; Lee, Y.; Khazatsky, A.; Manning, C. D.; and Finn, C. 2023. Detectgpt: Zero-shot machine-generated text detection using probability curvature. *arXiv preprint*.

Narayan, S.; Cohen, S. B.; and Lapata, M. 2018. Don't Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. In *Conference on Empirical Methods in Natural Language Processing*.

OpenAI. 2022. ChatGPT: Optimizing Language Models for Dialogue. http://web.archive.org/web/20230109000707/https://openai.com/blog/chatgpt/.

Park, R.; Rafailov, R.; Ermon, S.; and Finn, C. 2024. Disentangling length from quality in direct preference optimization. *arXiv preprint*.

Rafailov, R.; Sharma, A.; Mitchell, E.; Manning, C. D.; Ermon, S.; and Finn, C. 2024. Direct preference optimization: Your language model is secretly a reward model. *NeurIPS*, 36.

Sadasivan, V. S.; Kumar, A.; Balasubramanian, S.; Wang, W.; and Feizi, S. 2024. Can AI-Generated Text be Reliably Detected?

Solaiman, I.; Brundage, M.; Clark, J.; Askell, A.; Herbert-Voss, A.; Wu, J.; Radford, A.; Krueger, G.; Kim, J. W.; Kreps, S.; et al. 2019. Release strategies and the social impacts of language models. *arXiv preprint*.

Su, J.; Zhuo, T. Y.; Wang, D.; and Nakov, P. 2023. DetectLLM: Leveraging Log Rank Information for Zero-Shot Detection of Machine-Generated Text. *arXiv preprint*.

Tian, E.; and Cui, A. 2023. GPTZero: Towards detection of AI-generated text using zero-shot and supervised methods.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.-A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. 2023a. LLaMA: Open and Efficient Foundation Language Models.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023b. Llama 2: Open Foundation and Fine-Tuned Chat Models.

Verma, V.; Fleisig, E.; Tomlin, N.; and Klein, D. 2023. Ghostbuster: Detecting Text Ghostwritten by Large Language Models. *arXiv preprint*.

Wang, B.; and Komatsuzaki, A. 2021. GPT-J-6B: A 6 billion parameter autoregressive language model.

Wang, Y.; Mansurov, J.; Ivanov, P.; Su, J.; Shelmanov, A.; Tsvigun, A.; Whitehouse, C.; Afzal, O. M.; Mahmoud, T.; Sasaki, T.; et al. 2024. M4: Multi-Generator, Multi-Domain, and Multi-Lingual Black-Box Machine-Generated Text Detection. In *the European Chapter of the Association for Computational Linguistics*.

Wu, J.; Zhan, R.; Wong, D. F.; Yang, S.; Liu, X.; Chao, L. S.; and Zhang, M. 2024. Who Wrote This? The Key to Zero-Shot LLM-Generated Text Detection Is GECScore.

Yang, A.; Yang, B.; Hui, B.; Zheng, B.; Yu, B.; Zhou, C.; Li, C.; Li, C.; Liu, D.; Huang, F.; Dong, G.; Wei, H.; Lin, H.; Tang, J.; Wang, J.; Yang, J.; Tu, J.; Zhang, J.; Ma, J.; Yang, J.; Xu, J.; Zhou, J.; Bai, J.; He, J.; Lin, J.; Dang, K.; Lu, K.; Chen, K.; Yang, K.; Li, M.; Xue, M.; Ni, N.; Zhang, P.; Wang, P.; Peng, R.; Men, R.; Gao, R.; Lin, R.; Wang, S.; Bai, S.; Tan, S.; Zhu, T.; Li, T.; Liu, T.; Ge, W.; Deng, X.; Zhou, X.; Ren, X.; Zhang, X.; Wei, X.; Ren, X.; Liu, X.; Fan, Y.; Yao, Y.; Zhang, Y.; Wan, Y.; Chu, Y.; Liu, Y.; Cui, Z.; Zhang, Z.; Guo, Z.; and Fan, Z. 2024. Qwen2 Technical Report. arXiv:2407.10671.

Yang, X.; Cheng, W.; Wu, Y.; Petzold, L.; Wang, W. Y.; and Chen, H. 2023. DNA-GPT: Divergent N-Gram Analysis for Training-Free Detection of GPT-Generated Text. *arXiv preprint*.

Yuan, H.; Yuan, Z.; Tan, C.; Wang, W.; Huang, S.; and Huang, F. 2024. RRHF: Rank responses to align language models with human feedback. *NeurIPS*.

Zeng, C.; Tang, S.; Yang, X.; Chen, Y.; Sun, Y.; Li, Y.; Chen, H.; Cheng, W.; Xu, D.; et al. 2024. Improving Logits-based Detector without Logits from Black-box LLMs. *arXiv preprint*.

Zhang, Q.; Gao, C.; Chen, D.; Huang, Y.; Huang, Y.; Sun, Z.; Zhang, S.; Li, W.; Fu, Z.; Wan, Y.; et al. 2024. LLM-as-a-Coauthor: Can Mixed Human-Written and Machine-Generated Text Be Detected? *arXiv preprint*.

Zhu, B.; Yuan, L.; Cui, G.; Chen, Y.; Fu, C.; He, B.; Deng, Y.; Liu, Z.; Sun, M.; and Gu, M. 2023. Beat LLMs at Their Own Game: Zero-Shot LLM-Generated Text Detection via Querying ChatGPT. In *Empirical Methods in Natural Language Processing*.