MathGenBench: Benchmarking Detection of Machine-Generated Mathematical Text

Anonymous ACL submission

Abstract

The rapid advancement of large language models (LLMs) has heightened concerns about their misuse in generating deceptive mathematical content. To address the lack of specialized benchmarks for machine-generated mathematical text detection, we introduce MathGenBench, the first comprehensive benchmark targeting machine-generated mathematical text. Our benchmark integrates authentic human-written content from arXiv, Mathematics Stack Exchange (MSE), and Wikipedia with machine-generated samples produced by 10 leading language models. To simulate real-world adversarial scenarios, we employ various text manipulation strategies, including paraphrase attacks and perturbation attacks. Building upon the TOCSIN framework, we propose TOCSIN*, which enhances detection robustness through a learnable linear aggregation mechanism for token cohesiveness and zero-shot scores. Extensive experiments demonstrate TOCSIN*'s superiority over existing methods across different scenarios. This work provides critical tools for combating machine-generated mathematical text.

1 Introduction

Recent advanced LLMs, such as OpenAI o1 (Jaech et al., 2024) and Qwen 2.5 (Yang et al., 2024), have significantly elevated the quality of machine-generated text. These models achieve human-like fluency through enhanced reasoning capabilities and training on massive datasets (e.g., Qwen 2.5's 18T tokens). Consequently, their adoption has expanded across diverse domains, including advertising (Meguellati et al., 2024), journalism (Quinonez and Meij, 2024), creative writing (Gómez-Rodríguez and Williams, 2023), and code generation (Mu et al., 2024). However, the powerful generative capabilities of LLMs have also raised concerns about potential misuse, as their inherent limitations—including tendencies to fabricate facts (Ji et al., 2023), rely on outdated knowledge, and exhibit sensitivity to prompt phrasing—create vulnerabilities that could be exploited for spreading misinformation, enabling fraud (Ayoobi et al., 2023; Roy et al., 2024), generating spam (Mirsky et al., 2023), or facilitating academic misconduct (Kasneci et al., 2023). Furthermore, the growing practice of using machine-generated text in AI research training data risks creating feedback loops that could degrade data quality and diversity over time (Alemohammad et al., 2024).

To address these challenges, researchers have proposed benchmark datasets and evaluation methods to distinguish machine-generated text from human-written content. However, existing benchmarks often exhibit a paucity of samples from mathematical domains, limiting their effectiveness in evaluating detection performance in this specialized context. Mathematical texts differ significantly from general natural language due to their reliance on formal logical structures and the heavy use of symbolic notation. Existing detection methods can be broadly categorized into two categories: training-based methods and zero-shot methods. Training-based methods utilize classification models trained on corpora comprising both machine-generated and human-written text. While demonstrating strong detection performance following extensive training, they exhibit notable limitations, including poor interpretability, overfitting (Pu et al., 2023), and limited temporal adaptability. In contrast, zero-shot methods leverage statistical metrics like log-likelihood, entropy, token rank, and perplexity to perform thresholding-based classification. Specifically, TOCSIN (Ma and Wang, 2024) combines token cohesiveness and other zero-shot scores for machine-generated text detection. By directly utilizing pre-trained LLMs without fine-tuning, zero-shot methods circumvent domain adaptation challenges associated with model retraining.

To address the critical gap in domain-specific data insufficiency, we introduce MathGenBench (**Math**ematical Machine-**Gen**erated Text Detection **Bench**mark), a comprehensive benchmark of machine-generated text detection in mathematical domains. MathGenBench integrates multiple data sources, including arXiv, MSE and Wikipedia, and covers texts generated by 10 LLMs from the Qwen, Llama, and Mistral series. To simulate real-world detection scenarios, we employ various attack strategies, including DIPPER paraphraser, back-translation via Google Translate, and polishing using LLMs, as well as perturbation attacks at the character, word, and sentence levels. Based on this, we systematically evaluate the performance of mainstream zero-shot detectors on MathGenBench, providing an in-depth analysis of the detectors' performance differences across different data sources, LLMs, attack types, and text lengths.

Additionally, we enhance TOCSIN by introducing linear score aggregation, resulting in TOCSIN*. Experimental results show that TOCSIN* achieves competitive performance on MathGenBench under both white-box and black-box settings and demonstrates robustness against text attacks. By analyzing the parameters of TOCSIN*, we reveal that the importance of token cohesiveness increases with the scale of the LLMs. This finding provides a new perspective for understanding the detection mechanisms of texts generated by LLMs.

2 Related Work

2.1 Large Language Models

Research has demonstrated that increasing the parameter scale or training data volume of pre-trained language models generally leads to performance improvements on downstream tasks, a phenomenon known as the Scaling Law (Kaplan et al., 2020). For instance, GPT-3 has 175B parameters, while Google's subsequent PaLM scales up to 540B parameters. These large-scale pre-trained language models significantly outperform smaller counterparts in complex task scenarios. In zero-shot and few-shot learning tasks, large models can accomplish sophisticated tasks such as text summarization, translation, and question-answering without domain-specific training data or with minimal exemplars, producing outputs with enhanced accuracy and coherence. Notably, when pre-trained language models reach a critical scale threshold, they spontaneously exhibit novel characteristics or behaviors unseen in smaller models—a capability referred to as Emergent Abilities (Wei et al., 2022).

Since the release of ChatGPT (OpenAI, 2022) in November 2022, the development of LLMs has entered a phase of rapid advancement. Major technology companies worldwide have launched proprietary models, iteratively refining them through improved training frameworks and methodologies. Following GPT-3, OpenAI introduced GPT-4 (OpenAI, 2023) and GPT-4o (Hurst et al., 2024), enhancing multimodal task capabilities. In 2024, OpenAI unveiled the o1 (Ope, a) and o3 (Ope, b) series: o1 surpassed human PhD-level performance on benchmarks like GPQA Diamond and achieved 83% accuracy on AIME, while o3-mini achieved breakthroughs in programming, mathematical reasoning, and scientific inference, scoring 2727 points on Codeforces programming contests—approaching professional programmer proficiency. Since 2023, Meta has released the open-source Llama series including Llama 1 (Touvron et al., 2023a), Llama 2 (Touvron et al., 2023b), and Llama 3 (Dubey et al., 2024), each iteration introducing architectural innovations to improve performance and versatility. DeepSeek released the DeepSeek-V3 (Liu et al., 2024) model, a 671B-parameter Mixture-of-Experts (MoE) architecture with 37B activated parameters, pre-trained on 14.8T tokens. It demonstrates notable advancements in knowledge representation, long-text processing, coding, mathematics, and Chinese language tasks, achieving a 3× speed improvement (60 TPS) over its V2.5 version.

2.2 Machine-Generated Text Detection

Current detection methods can be devided in to two categories, i.e., training-based methods and zero-shot methods. In this paper we consider eight zero-shot detection methods, including Log-Likelihood, Rank, Log-Rank, Entropy, LRR, Fast-DetectGPT, Binoculars and TOCSIN.

- **Log-Likelihood** (Solaiman et al., 2019): This approach leverages a language model to calculate the token-wise log probability of a given input text. Higher values indicate a higher probability of being generated by a large language model.
- **Rank** (Gehrmann et al., 2019): This metric calculates the average rank of token probabilities in the model's output distribution. Lower scores suggest text is more likely machine-generated.

- **Log-Rank** (Solaiman et al., 2019): Similar to Rank metric, Log-Rank uses logarithmic rank values instead of raw ranks for calculation.
- **Entropy** (Ippolito et al., 2020): This metric calculates the average entropy of token probability distributions conditioned on preceding context. Machine-generated text typically exhibits lower entropy values.
- **LRR** (Su et al., 2023): Log-**L**ikelihood Log-**R**ank **R**atio (LRR) combines log-likelihood and log-rank through division. It outperforms DetectGPT (Mitchell et al., 2023) by capturing complementary features.
- **Fast-DetectGPT** (Bao et al., 2024): Fast-DetectGPT enhances DetectGPT by using a more efficient sampling strategy. It proposed a metric named conditional probability curvature and observed that machine-generated text tends to have higher conditional probability curvature.
- **Binoculars** (Hans et al., 2024): Binoculars calculates a detection score by contrasting the perplexity of a text under observer model with the cross-perplexity computed using observer model and performer model.
- **TOCSIN** (Ma and Wang, 2024): TOCSIN leverages token cohesiveness as a plug-and-play module to improve existing zero-shot detectors, based on the observation that machine-generated text tends to exhibit higher token cohesiveness.

3 MathGenBench

Existing benchmarks for detecting machine-generated text struggle with mathematical content due to insufficient domain-specific samples and limited coverage of complex notation and domain-specific reasoning patterns. To address this gap, we introduce MathGenBench, a specialized benchmark combining three critical dimensions: (1) human-authored mathematical texts from moderated sources (arXiv, MSE, and Wikipedia), (2) machine-generated content from 10 leading LLMs, and (3) adversarially augmented samples simulating real-world evasion tactics. This resource enables comprehensive evaluation of detection robustness, while providing granular insights into model-specific generation patterns and attack vulnerabilities. The remainder of this section is organized as follows: We first present the design framework of MathGenBench, followed by dataset statistics and analysis.

3.1 Benchmark Construction Framework

Data Curation Human-written texts in MathGenBench are from three rigorously moderated sources: 500 academic abstracts from arXiv Mathematics (subject to disciplinary moderation and endorsement policies), 500 question-answer pairs from MSE (peer-reviewed through community voting and expert moderation), and 500 encyclopedic summaries from Wikipedia mathematics portals (maintained through citation-based verification and editorial oversight). The institutional governance mechanisms inherent to these platforms—spanning academic validation, collaborative quality control, and verifiability standards—minimize the inclusion of AI-generated content while ensuring authoritative representation of mathematical discourse.

Models To construct the machine-generated text corpus, we selected 10 leading open-source models from Qwen, Llama, and Mistral series as of November 2024. Machine-generated texts are generated by running inference on these models. For more details on the LLMs and generation protocol, please refer to Appendix A.

Adversarial Augmentation To simulate complex real-world detection scenarios and evaluate the robustness of detection methodologies, following DetectRL (Wu et al., 2024), we implemented adversarial augmentation with paraphrase attacks and perturbation attacks.

Paraphrase attacks are designed to preserve semantic integrity while altering surface-level expressions. We employed three approaches: the DIPPER paraphraser (Krishna et al., 2023), back-translation via Google Translate, and polish using LLMs. More details of polish paraphrase can be found in Appendix A.3.

Perturbation attacks focus on inducing misclassification through minimal modifications, utilizing three established frameworks targeting different linguistic granularities: DeepWordBug (Gao et al., 2018) for character-level transformations through critical token identification via scoring functions, TextFooler (Jin et al., 2020) for context-aware synonym substitution with linear time complexity, and TextBugger (Li et al., 2019) as a unified framework generating visually/semantically consistent adversarial examples for deep learning-based text

understanding systems. These methodologies collectively enabled comprehensive evaluation of detection robustness across lexical, syntactic, and semantic dimensions while preserving real-world operational validity.

3.2 Benchmark Statistics

The statistical breakdown of dataset categories is presented in table 1. The corpus contains 10,500 human-authored texts, comprising 1,500 original samples and 9,000 instances derived through paraphrasing and perturbation attacks. The distribution of machine-generated texts mirrors this structure: for each model, 1,500 original generations and 9,000 attack-modified samples are included, with a total of 10 distinct models employed for text generation. Figure 1 illustrates the word count distributions of original texts compared to Qwen-14B generated outputs. Cross-source analyses reveal minimal disparities in word count distributions between original and generated texts: arXiv texts predominantly concentrate within 250 words (93.2% of samples), while MSE texts exhibit broader distribution spans with 8.2% of samples exceeding 1,000 words. Further granularity is provided in fig. 2, which visualizes model-specific word count distributions across different data sources.

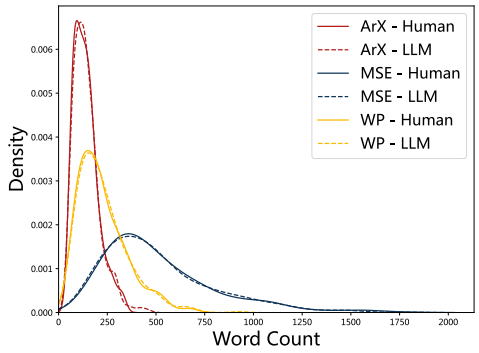

Figure 1: Word count distribution of original text and Qwen-14B generated text

4 Method

4.1 TOCSIN

TOCSIN (Ma and Wang, 2024) introduces BARTScore (Yuan et al., 2021) to measure token cohesiveness, achieving enhanced detection performance through integration with other zero-shot detection methods including Likelihood, LogRank, LRR, and Fast-DetectGPT. The key assumption of TOCSIN is that LLM-generated text typically exhibits higher token cohesiveness compared to human-written text.

Token Cohesiveness Token cohesiveness quantifies the semantic difference between candidate text and its perturbed variant. Higher token cohesiveness indicates stronger semantic coherence among consecutive tokens. Ma et al. (Ma and Wang, 2024) hypothesize that text generated by LLMs and human-written text exhibit distinct characteristics in token cohesiveness. LLMs inherently generate tokens sequentially based on preceding context, which naturally strengthens semantic dependencies between adjacent tokens. In contrast, human writing involves greater lexical flexibility and is subject to subjective stylistic choices without explicit sequential constraints. Formally, token cohesiveness is defined as:

$$u(x) \triangleq \mathbb{E}(\mathrm{DIFF}(x, \tilde{x})),$$

where $\mathrm{DIFF}(\cdot,\cdot)$ denotes a semantic difference metric and $\mathbb{E}(\cdot)$ represents expectation operator. Given input text x, its token cohesiveness can be empirically estimated through perturbation analysis: generating N perturbed variants $\{\tilde{x}^i\}_{i=1}^N$ and computing the sample mean of semantic differences:

$$\hat{u}(x) = \sum_{i=1}^{N} \frac{\mathrm{DIFF}(x, \tilde{x}^i)}{N}. \quad (1)$$

In the TOCSIN framework, $\mathrm{DIFF}(\cdot,\cdot)$ is implemented using the negative BARTScore, which calculates semantic similarity through conditional log-likelihood:

$$\mathrm{BARTScore}(x, \tilde{x}) = \sum_{j=1}^{k} \log p_\phi(x_j | x_{<j}, \tilde{x}),$$

where ϕ represents the parameters of the BART model.

4.2 TOCSIN*

We see an opportunity to improve TOCSIN's score aggregation mechanism: it uses the following formula

$$w(x) = \mathrm{e}^{\mathrm{sgn}(v(x))\hat{u}(x)} v(x)$$

to combine the empirical token cohesiveness $\hat{u}(x)$ and the score obtained by some zero-shot method. This formula is not fully discussed in (Ma and Wang, 2024) and lacks weight control on scores from two distinct sources.

Data Source	Attack Types	-	Paraphrase Attacks			Perturbation Attacks			Total
			BP	DP	PP	CP	WP	SP	
ArX	Human	500	500	500	500	500	500	500	3,500
	LLMs	5,000	5,000	5,000	5,000	5,000	5,000	5,000	35,000
MSE	Human	500	500	500	500	500	500	500	3,500
	LLMs	5,000	5,000	5,000	5,000	5,000	5,000	5,000	35,000
WP	Human	500	500	500	500	500	500	500	3,500
	LLMs	5,000	5,000	5,000	5,000	5,000	5,000	5,000	35,000
Total	-	16,500	16,500	16,500	16,500	16,500	16,500	16,500	115,500

Table 1: Benchmark statistics; ArX: arXiv, MSE: Mathematics Stack Exchange, WP: Wikipedia; LLMs refers to the collective term for all models selected in the paper; BP: Back-translation Paraphrase, DP: DIPPER Paraphrase, PP: Polish Paraphrase, CP: Character-level Perturbation, WP: Word-level Perturbation, SP: Sentence-level Perturbation.

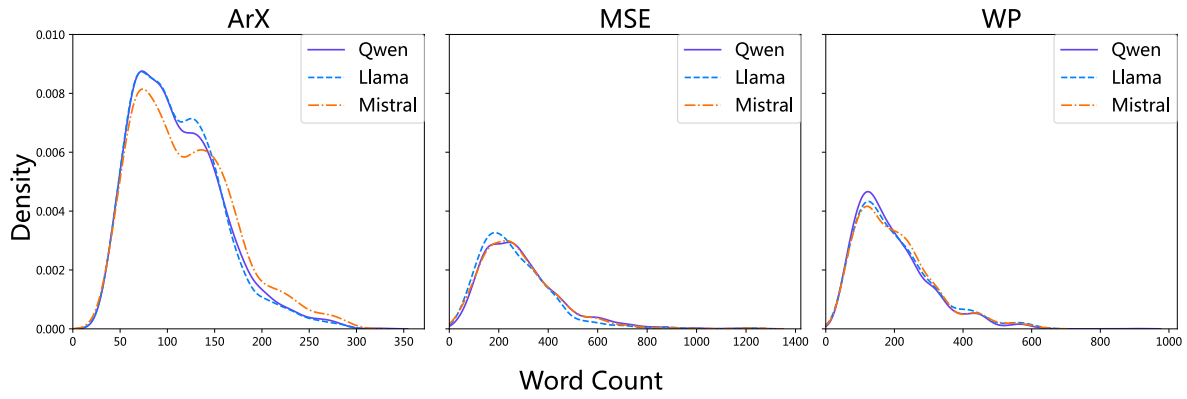

Figure 2: Word count distribution of generated texts across model series and data sources

Here, we propose a simple linear score aggregation mechanism as follows:

$$w(x) = \alpha\hat{u}(x) + \beta v(x), \tag{2}$$

where α and β are learnable parameters obtained via logistic regression on training data. This approach allows adaptive weight assignment between the empirical token cohesiveness score $\hat{u}(x)$ and the zero-shot score $v(x)$, ensuring appropriate contributions from each source to the final detection metric. We name the framework using the above new detection metric TOCSIN*, and its framework is shown in fig. 3. The effectiveness of (2) will be verified in the next section.

5 Experiments

5.1 Detectors Configuration

We conducted evaluations on multiple representative zero-shot methods, including five logits-based approaches (Log-Likelihood, Rank, Log-Rank, Entropy, and LRR), perturbation-based methods (Fast-DetectGPT and TOCSIN), Binoculars, and our proposed improved version of TOCSIN (TOCSIN*). To streamline the experimental setup, in the black-box scenario, both the sampling model and scoring model in Fast-DetectGPT were implemented using Neo-2.7 (Black et al., 2021). Similarly, all other logits-based methods employed Neo-2.7 as the proxy model. BARTScore in TOCSIN and TOCSIN* was computed using BART-base(Lewis et al., 2020). According to the experimental results from Binoculars (Hans et al., 2024), the observer model and performer model in Binoculars were implemented using Falcon-7B and Falcon-7B-Instruct (Almazrouei et al., 2023) respectively.

5.2 Main Results

Table 4 and table 5 present the evaluation results of detectors under white-box and black-box settings, respectively.

Under the white-box setting, in terms of detection performance averaged across all models, our proposed TOCSIN* achieves the highest average AUROC on the arXiv dataset (corresponding to the dataset generated from arXiv data source,

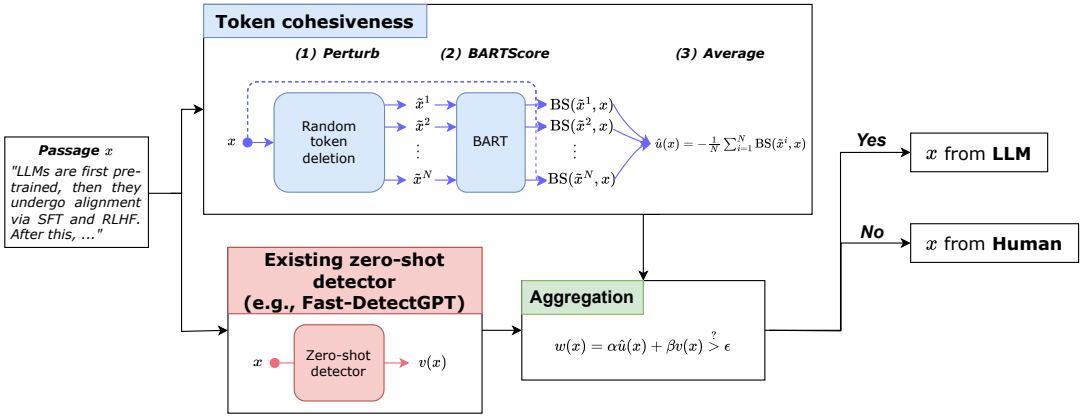

Figure 3: TOCSIN* framework

similarly for MSE and Wikipedia datasets) and Wikipedia dataset. While Binoculars performs best on the MSE dataset, TOCSIN* closely follows with superior performance compared to TOCSIN. Notably, TOCSIN* demonstrates the most significant improvements over TOCSIN on three specific dataset-model combinations: (MSE, Mistral-7B), (Wikipedia, Qwen-14B), (Wikipedia, Mistral-8B), and (Wikipedia, Mistral-12B), achieving AUROC gains of 5.77%, 5.47%, 8.41% and 5.50%, respectively. However, TOCSIN shows no advantage over Fast-DetectGPT, with lower average AUROC values on both arXiv and MSE datasets. Notably, Binoculars outperforms all other methods on the MSE dataset without requiring access to the source model. Among other zero-shot methods, LRR and Log-Rank exhibit the most competitive performance, ranking high in AUROC averages across all three datasets, while Entropy performs worst with AUROC values below 0.5 across all datasets.

Regarding dataset characteristics, from the perspective of data source, arXiv demonstrates the lowest detection difficulty, whereas MSE presents the greatest challenge. This may be attributed to the highly specialized and logically structured nature of human-written texts in arXiv (being article abstracts with concise language), contrasting with the general-knowledge-oriented and potentially redundant outputs from LLMs. For the MSE dataset, the more complete prompts provided during text generation likely enable better topic comprehension by language models, resulting in outputs closer to human-written texts. From the perspective of generative model, Llama-series models exhibit the lowest detection difficulty, with Fast-DetectGPT, TOCSIN, and TOCSIN* achieving AUROC values above 99% on both arXiv and Wikipedia datasets. Interestingly, within each model family, detection performance (AUROC) generally decreases with increased parameter scale. This phenomenon might be explained by larger models' enhanced capacity to internalize diverse training corpus knowledge, thereby generating more indistinguishable texts.

Under the black-box setting, regarding detection methodologies, the performance of all detection methods generally declined. On the arXiv and Wikipedia datasets, TOCSIN* demonstrated significantly greater advantages compared to TOCSIN, achieving average AUROC improvements of 8.77% and 13.44%, respectively. Similar to the white-box setting, TOCSIN* exhibited slightly lower average AUROC than Fast-DetectGPT on the MSE dataset but still outperformed TOCSIN. Binoculars performed second only to TOCSIN* on the arXiv and Wikipedia datasets. Among other zero-shot methods, LRR and Log-Rank achieved the best performance, consistently ranking among the top two in terms of average AUROC across all three datasets.

5.3 Impact of Adversarial Attack

We tested the detectors on datasets generated by Qwen-series models rewritten with various attack methods. The results are shown in fig. 4.

Under the white-box setting, paraphrasing attacks using DIPPER and polishing significantly impacted detector performance. All detectors achieved average AUROC values below 0.6 on datasets modified by these two attacks. While back-translation had relatively smaller effects compared to the former methods, it still caused a 21.97% decrease in average AUROC for Fast-DetectGPT.

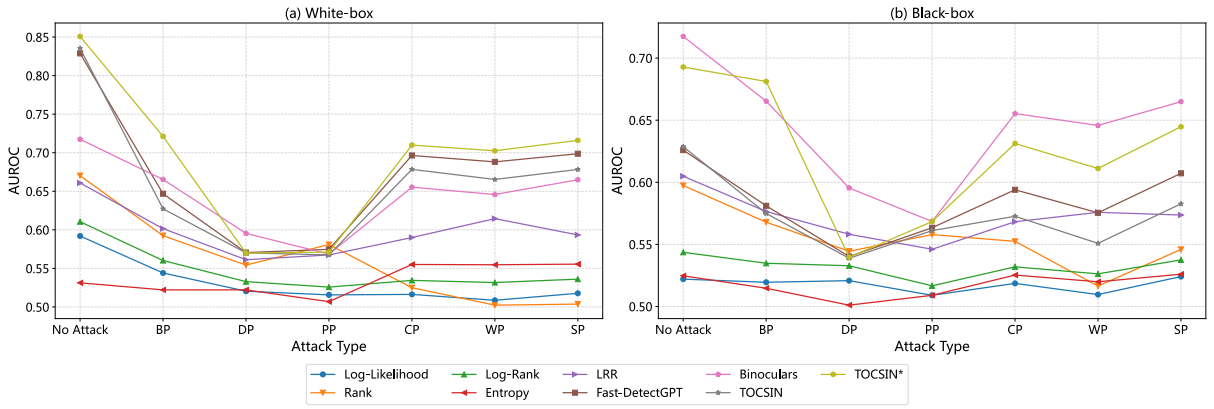

Figure 4: Average AUROC across data sources and generative models under white-box and black-box settings

Although TOCSIN performed slightly better than Fast-DetectGPT on original data, its average AUROC under character-, word-, and sentence-level perturbation attacks was notably lower. TOCSIN* outperformed other detectors on all attacks except DIPPER and polishing paraphrasing, surpassing Fast-DetectGPT and Binoculars by 11.52% and 8.4% in average AUROC, respectively, on back-translation attacked datasets.

Under the black-box setting, all detectors showed decreased average AUROC compared to white-box scenario. Similar to white-box results, DIPPER and polishing paraphrasing attacks remained highly impactful, with average AUROC values below 0.57 for all detectors except Binoculars on these modified datasets. Notably, TOCSIN* demonstrated more pronounced advantages over TOCSIN and Fast-DetectGPT in black-box scenario. Under character-, word-, and sentence-level perturbations, TOCSIN* consistently outperformed Fast-DetectGPT (which in turn surpassed TOCSIN), with an average AUROC difference exceeding 0.025. Remarkably, TOCSIN* exhibited strong robustness against back-translation paraphrasing attacks, with only a 1.67% decrease in average AUROC.

5.4 Impact of Text Length

Studies by Verma (Verma et al., 2024) and Mao (Mao et al., 2024) have shown that shorter texts are more challenging to detect. To validate this, we performed truncation experiments on datasets generated by Qwen-series models with varying text lengths (measured by word count). The results are presented in fig. 5.

Under the white-box setting, all detectors except Entropy exhibited improved performance with longer input texts. For truncations under 800 words, LRR achieved higher average AUROC than Rank, while Rank outperformed LRR when no truncation was applied. The performance gap among Fast-DetectGPT, TOCSIN, and TOCSIN* widened progressively with increasing truncation length. Notably, TOCSIN* achieved the highest average AUROC (surpassing both Fast-DetectGPT and TOCSIN) when no truncation was performed.

The black-box setting revealed similar trends, with all methods except Entropy benefiting from longer text inputs. However, the performance differences between TOCSIN* and other methods (Fast-DetectGPT/TOCSIN) were more pronounced compared to white-box scenarios. Specifically, TOCSIN* underperformed relative to Fast-DetectGPT and TOCSIN when input length was truncated to less or equal than 100 words. As truncation length increased to 200 words, TOCSIN* gradually closed the performance gap and ultimately achieved higher average AUROC than both Fast-DetectGPT and TOCSIN. Notably, Binoculars maintained the highest performance across all truncation lengths, with its superiority margin over TOCSIN* initially widening and then narrowing as inputs approached full length.

5.5 Parameter Interpretability Analysis

TOCSIN* employs training data to learn the score aggregation parameters α and β. After standardizing the features, the absolute values of the logistic regression coefficients reflect the importance of each feature. Figure 6 illustrates the absolute coefficient ratios ($\frac{|\beta|}{|\alpha|}$) across datasets and models. The results show that, except for the (Wikipedia, Llama) dataset, the ratio decreases with increasing model parameter size within each model series, in-

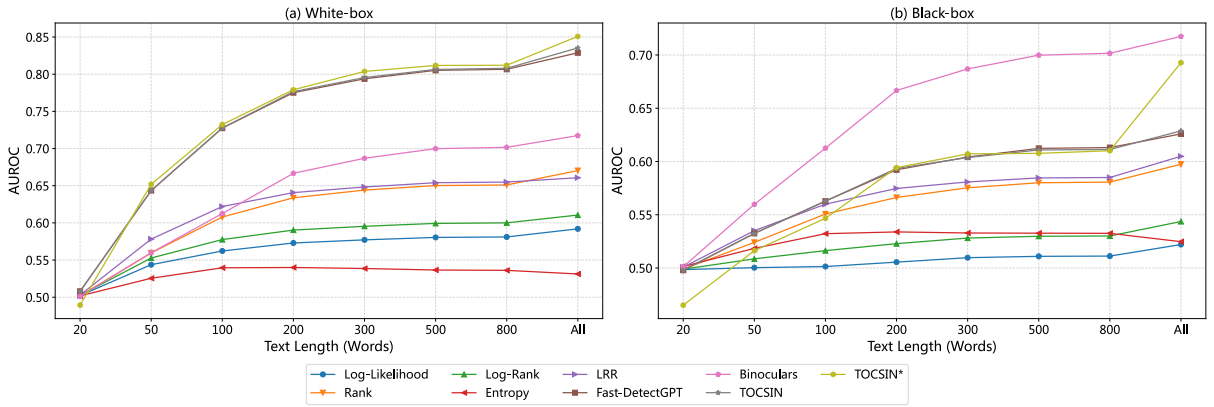

Figure 5: Average AUROC changes as text length varies under white-box and black-box settings

dicating a gradual rise in the importance of token cohesiveness.

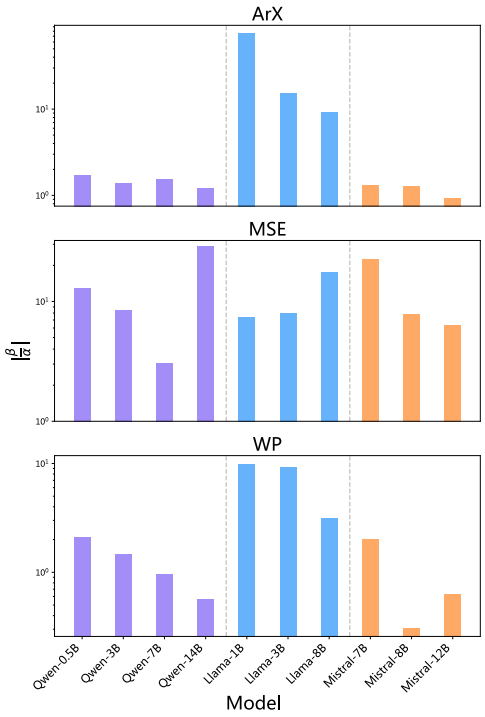

Figure 6: Absolute logistic regression coefficient ratios ($|\frac{\beta}{\alpha}|$) in TOCSIN*

6 Conclusion

This paper tackles the critical challenge of detecting LLM-generated text through two key contributions: (1) the construction of MathGenBench, a large-scale mathematical domain benchmark encompassing 115,500 annotated samples across diverse sources (arXiv, Mathematics Stack Exchange and Wikipedia), multiple LLM families (Qwen, Llama, and Mistral series), and adversarial attack variants; and (2) the development of TOCSIN*—an enhanced zero-shot detection framework. Through systematic evaluation, we reveal critical insights: detection difficulty increases as the scale of LLMs grows; MSE datasets exhibit higher detection difficulty than arXiv and Wikipedia corpora; Llama-derived text shows the lowest detection resistance; input length positively correlates with performance, while paraphrasing/perturbation attacks severely degrade detection reliability. TOCSIN*, combining token cohesiveness and other zero-shot methods through linear score aggregation, demonstrates the best detection performance across all scenarios.

Limitations

This work has two limitations: First, due to cost constraints, the machine-generated text in our benchmark does not include outputs from the latest closed-source LLMs (e.g., GPT-4o, Claude 3.7), which may limit the generalization of detection performance to cutting-edge proprietary models. Second, while TOCSIN* demonstrates superior performance over existing zero-shot methods on MathGenBench, its effectiveness in non-mathematical domains remains to be thoroughly validated.

References

a. OpenAI o1 Hub | OpenAI. https://openai.com/o1/.

b. OpenAI o3-mini. https://openai.com/index/openai-o3-mini/.

Meta AI. 2024a. Llama 3.2: Revolutionizing edge AI and vision with open, customizable models. https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

Mistral AI. 2024b. Mistral NeMo. https://mistral.ai/news/mistral-nemo/.

Mistral AI. 2024c. Un Ministral, des Ministraux. https://mistral.ai/news/ministraux/.

Sina Alemohammad, Josue Casco-Rodriguez, Lorenzo Luzi, Ahmed Imtiaz Humayun, Hossein Babaei, Daniel LeJeune, Ali Siahkoohi, and Richard G. Baraniuk. 2024. Self-consuming generative models go MAD. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. ICLR.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of open language models. *CoRR*, abs/2311.16867.

Navid Ayoobi, Sadat Shahriar, and Arjun Mukherjee. 2023. The looming threat of fake and llm-generated linkedin profiles: Challenges and opportunities for detection and prevention. In *Proceedings of the 34th ACM Conference on Hypertext and Social Media, HT 2023, Rome, Italy, September 4-8, 2023*, pages 38:1–38:10. ACM.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. 2024. Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Sid Black, Gao Leo, Phil Wang, Connor Leahy, and Stella Biderman. 2021. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, and 82 others. 2024. The llama 3 herd of models.

Ji Gao, Jack Lanchantin, Mary Lou Soffa, and Yanjun Qi. 2018. Black-box generation of adversarial text sequences to evade deep learning classifiers. In *2018 IEEE Security and Privacy Workshops, SP Workshops 2018, San Francisco, CA, USA, May 24, 2018*, pages 50–56. IEEE Computer Society.

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

Carlos Gómez-Rodríguez and Paul Williams. 2023. A confederacy of models: a comprehensive evaluation of llms on creative writing. In *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 14504–14528. Association for Computational Linguistics.

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 *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.

Aaron Hurst, Adam Lerer, Adam P. Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, Aleksander Madry, Alex Baker-Whitcomb, Alex Beutel, Alex Borzunov, Alex Carney, Alex Chow, Alex Kirillov, Alex Nichol, Alex Paino, and 79 others. 2024. Gpt-4o system card. abs/2410.21276.

Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. 2020. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 1808–1822. Association for Computational Linguistics.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, Alex Iftimie, Alex Karpenko, Alex Tachard Passos, Alexander Neitz, Alexander Prokofiev, Alexander Wei, Allison Tam, Ally Bennett, Ananya Kumar, and 80 others. 2024. Openai o1 system card. abs/2412.16720.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12):248:1–248:38.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is BERT really robust? A strong baseline for natural language attack on text classification and entailment. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8018–8025. AAAI Press.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models.

Enkelejda Kasneci, Kathrin Sessler, Stefan Küchemann, Maria Bannert, Daryna Dementieva, Frank Fischer, Urs Gasser, Georg Groh, Stephan Günnemann, Eyke Hüllermeier, Stephan Krusche, Gitta Kutyniok, Tilman Michaeli, Claudia Nerdel, Jürgen Pfeffer, Oleksandra Poquet, Michael Sailer, Albrecht Schmidt, Tina Seidel, and 4 others. 2023. Chatgpt for good? on opportunities and challenges of large language models for education. *Learning and Individual Differences*, 103:102274.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. 2019. Textbugger: Generating adversarial text against real-world applications. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, and 80 others. 2024. Deepseek-v3 technical report. abs/2412.19437.

Shixuan Ma and Quan Wang. 2024. Zero-shot detection of LLM-generated text using token cohesiveness. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17538–17553, Miami, Florida, USA. Association for Computational Linguistics.

Chengzhi Mao, Carl Vondrick, Hao Wang, and Junfeng Yang. 2024. Raidar: Generative AI detection via rewriting. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Elyas Meguellati, Lei Han, Abraham Bernstein, Shazia Sadiq, and Gianluca Demartini. 2024. How good are llms in generating personalized advertisements? In *Companion Proceedings of the ACM on Web Conference 2024, WWW 2024, Singapore, Singapore, May 13-17, 2024*, pages 826–829. ACM.

Yisroel Mirsky, Ambra Demontis, Jaidip Kotak, Ram Shankar, Gelei Deng, Liu Yang, Xiangyu Zhang, Maura Pintor, Wenke Lee, Yuval Elovici, and Battista Biggio. 2023. The threat of offensive AI to organizations. *Comput. Secur.*, 124:103006.

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, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 24950–24962. PMLR.

Fangwen Mu, Lin Shi, Song Wang, Zhuohao Yu, Binquan Zhang, Chenxue Wang, Shichao Liu, and Qing Wang. 2024. Clarifygpt: A framework for enhancing llm-based code generation via requirements clarification. *Proc. ACM Softw. Eng.*, 1(FSE):2332–2354.

OpenAI. 2022. ChatGPT: Optimizing language models for dialogue. *https: //openai.com/blog/chatgpt*.

OpenAI. 2023. GPT-4 technical report. abs/2303.08774.

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 *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, May 21-25, 2023*, pages 1613–1630. IEEE.

Claudia Quinonez and Edgar Meij. 2024. A new era of ai-assisted journalism at bloomberg. *AI Mag.*, 45(2):187–199.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*.

Sayak Saha Roy, Poojitha Thota, Krishna Vamsi Naragam, and Shirin Nilizadeh. 2024. From chatbots to phishbots?: Phishing scam generation in commercial large language models. In *IEEE Symposium on Security and Privacy, SP 2024, San Francisco, CA, USA, May 19-23, 2024*, pages 36–54. IEEE.

Irene Solaiman, Miles Brundage, Jack Clark, Amanda Askell, Ariel Herbert-Voss, Jeff Wu, Alec Radford, and Jasmine Wang. 2019. Release strategies and the social impacts of language models.

Jinyan Su, Terry Yue 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, Singapore, December 6-10,*

2023, pages 12395–12412. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023b. Llama 2: Open foundation and fine-tuned chat models. abs/2307.09288.

Vivek Verma, Eve Fleisig, Nicholas Tomlin, and Dan Klein. 2024. Ghostbuster: Detecting text ghostwritten by large language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 1702–1717. Association for Computational Linguistics.

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

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022. Emergent abilities of large language models. *Transactions on Machine Learning Research*, 2022.

Junchao Wu, Runzhe Zhan, Derek F. Wong, Shu Yang, Xinyi Yang, Yulin Yuan, and Lidia S. Chao. 2024. Detectrl: Benchmarking llm-generated text detection in real-world scenarios. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report.

Weizhe Yuan, Graham Neubig, and Pengfei Liu. 2021. Bartscore: Evaluating generated text as text generation. In *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 27263–27277.

A Data Collection

A.1 Model Inventory

We selected prominent open-source models with top rankings in large language model benchmarks as of November 2024. Table 2 provides detailed specifications of the source models, including their repository paths and parameter counts. All models are run locally on a single NVIDIA A6000 GPU (48GB).

A.2 Generation Protocol

The machine-generated text in the dataset was produced by running inference on models downloaded from Hugging Face. For arXiv and Wikipedia corpora, which contain continuous text passages, we get machine-generated text by prompting the model to with the first 30 tokens of the source text. In contrast, the MSE dataset's question-answer structure required a different approach: source questions were directly employed as prompts to generate machine-generated answers, maintaining the original task's question-answering paradigm.

A.3 Polish Paraphrase

Polishing with LLMs has emerged as a widely adopted paraphrasing technique. The prompt templates employed in the process are shown in Table 3. For arXiv and Wikipedia corpora, the {prompt} field is the complete pre-paraphrase text, while for the MSE dataset, the {question} and {prompt} fields correspond to the question and answer text respectively.

B Main Results

The results of white-box and black-box settings are shown in Table 4 and Table 5, respectively.

C Ablation Study

C.1 Ablation on Question Texts in MSE Dataset

During the creation of the MSE dataset, we used question texts as prompts to generate corresponding answers through LLMs, thereby obtaining machine-generated texts. To investigate the impact of question inclusion, we compared detection performance with and without question texts on the MSE dataset. The results are presented in table 6.

For Fast-DetectGPT and TOCSIN, including question texts yielded better detection performance in the majority of cases. This pattern also held

Model	Model File	Parameters
Qwen-0.5B (Yang et al., 2024)	Qwen/Qwen2.5-0.5B	494M
Qwen-3B	Qwen/Qwen2.5-3B	3.09B
Qwen-7B	Qwen/Qwen2.5-7B	7.62B
Qwen-14B	Qwen/Qwen2.5-14B	14.8B
Llama-1B (AI, 2024a)	meta-llama/Llama-3.2-1B	1.24B
Llama-3B	meta-llama/Llama-3.2-3B	3.21B
Llama-8B (Dubey et al., 2024)	meta-llama/Llama-3.1-8B	8.03B
Mistral-7B (Jiang et al., 2023)	mistralai/Mistral-7B-Instruct-v0.3	7.25B
Mistral-8B (AI, 2024c)	mistralai/Ministral-8B-Instruct-2410	8.02B
Mistral-12B (AI, 2024b)	mistralai/Mistral-Nemo-Instruct-2407	12.2B

Table 2: Details of the source models used to produce machine-generated text

Data Source	Polish Prompt Template
ArX	Given the article abstract, polish the writing to meet the academic abstract style and mathematical writing style with {sentences_num} sentences, improve the spelling, grammar, clarity, concision and overall readability: abstract: {prompt} polished abstract:
MSE	Given a Q&A pair, polish the answer to meet mathematical writing style with {sentences_num} sentences, improve the spelling, grammar, clarity, logical flow, concision, and overall readability: question: {question} original answer: {prompt} polished answer:
WP	Given the wikipedia page summary, polish the writing to meet the wikipedia page style and mathematical writing style with {sentences_num} sentences, improve the spelling, grammar, clarity, concision and overall readability: summary: {prompt} polished summary:

Table 3: Polish prompt templates

true for Log-Likelihood and Log-Rank methods. Conversely, for Rank, Entropy, LRR, and Binoculars, excluding question texts produced superior results. It is noteworthy that all main detection results reported for the MSE dataset in this study incorporated question texts during analysis.

C.2 Ablation on Proxy Models Selection

Bao (Bao et al., 2024) investigated the impact of different sampling models on detection performance under white-box settings. To analyze this effect, we compared the performance of detectors using various proxy models on the Qwen dataset. As shown in table 7, the optimal proxy models for achieving the highest average AUROC varied across detectors.

For Fast-DetectGPT and TOCSIN, NEO-2.7 emerged as the best-performing proxy model, while GPT-2 demonstrated superior results for Log-Likelihood, Log-Rank, and LRR. Notably, NEO-2.7 consistently achieved the highest or second-highest AUROC values across all data sources and detectors compared to alternative proxy models. Based on these findings, NEO-2.7 was selected as the default proxy model for all black-box experiments in this study.

Data Source	Model / Method	Qwen-0.5B	Qwen-3B	Qwen-7B	Qwen-14B	Llama-1B	Llama-3B	Llama-8B	Mistral-7B	Mistral-8B	Mistral-12B	Avg.
ArX	Log-Likelihood	0.7636	0.7069	0.6989	0.7035	0.9491	0.9403	0.9338	0.8120	0.7935	0.7646	0.8066
	Rank	0.7761	0.7303	0.7072	0.7030	0.7378	0.7203	0.7068	0.7492	0.7286	0.7319	0.7291
	Log-Rank	0.8051	0.7395	0.7265	0.7230	0.9658	0.9586	0.9535	0.8269	0.8155	0.7888	0.8303
	Entropy	0.4704	0.5000	0.4945	0.4847	0.3455	0.3596	0.3741	0.4081	0.3972	0.4092	0.4243
	LRR	0.8744	0.7983	0.7805	0.7522	0.9785	0.9690	0.9672	0.8265	0.8520	0.8289	0.8628
	Fast-DetectGPT	0.9261	0.9136	0.9133	0.8960	**0.9985**	**0.9986**	0.9979	**0.9565**	0.9454	0.9244	0.9470
	Binoculars	0.8129	0.7543	0.7121	0.6471	0.9969	0.9938	0.9835	0.7797	0.8400	0.7527	0.8273
	TOCSIN	**0.9287**	0.9146	0.9122	0.8941	0.9982	0.9985	0.9981	0.9463	0.9405	0.9248	0.9456
	TOCSIN*	0.9260	**0.9331**	**0.9195**	**0.9197**	0.9954	0.9966	**0.9983**	0.9460	**0.9716**	**0.9425**	**0.9549**
MSE	Log-Likelihood	0.5442	0.5666	0.6043	0.5455	0.7491	0.7196	0.7382	0.7481	0.6224	0.5777	0.6416
	Rank	0.5989	0.5890	0.5801	0.5734	0.6050	0.5884	0.5949	0.6337	0.5774	0.5500	0.5891
	Log-Rank	0.5558	0.5733	0.6048	0.5515	0.7526	0.7262	0.7472	0.7265	0.6182	0.5815	0.6437
	Entropy	0.5108	0.4731	0.4219	0.4902	0.3542	0.3866	0.3753	0.3651	0.4232	0.4763	0.4277
	LRR	0.5992	0.5856	0.5778	0.5714	0.7062	0.7109	0.7245	0.5567	0.5755	0.5882	0.6196
	Fast-DetectGPT	0.7576	0.7257	0.6790	0.6873	0.9287	0.9173	**0.9333**	**0.9195**	0.7652	0.7347	0.8048
	Binoculars	0.6901	0.7205	**0.7650**	0.6817	0.9215	0.9196	0.9146	0.8685	**0.8157**	**0.7673**	**0.8065**
	TOCSIN	0.7600	0.7261	0.6769	0.6994	0.9130	0.9074	0.9256	0.8572	0.7247	0.7326	0.7923
	TOCSIN*	**0.7602**	**0.7351**	0.7086	**0.7020**	**0.9380**	**0.9217**	0.9292	0.9149	0.7247	0.6858	0.8020
WP	Log-Likelihood	0.6676	0.4895	0.4352	0.3776	0.9726	0.9479	0.7615	0.7955	0.6624	0.6251	0.6735
	Rank	0.8382	0.7055	0.6476	0.5944	0.8232	0.7931	0.6984	0.7570	0.7488	0.6990	0.7305
	Log-Rank	0.7164	0.5042	0.4453	0.3807	0.9850	0.9689	0.8055	0.8001	0.6869	0.6498	0.6943
	Entropy	0.5664	0.6422	0.6439	0.6772	0.2018	0.2583	0.4300	0.4898	0.5001	0.5520	0.4962
	LRR	0.8339	0.5819	0.5137	0.4601	0.9914	0.9868	0.8848	0.7652	0.7406	0.7157	0.7474
	Fast-DetectGPT	0.9495	0.8933	0.8455	0.7589	0.9975	0.9975	0.9680	0.9907	0.8488	0.9030	0.9153
	Binoculars	0.7883	0.7070	0.7014	0.6285	0.9976	0.9915	0.9200	0.8581	0.8432	0.7584	0.8194
	TOCSIN	**0.9572**	0.9033	0.8610	0.7902	0.9970	0.9977	0.9725	0.9839	0.8720	0.8996	0.9235
	TOCSIN*	0.9449	**0.9206**	**0.8949**	**0.8449**	**0.9997**	**0.9998**	**0.9851**	**0.9951**	**0.9561**	**0.9546**	**0.9496**

Table 4: AUROC across data sources and generative models under white-box setting

Data Source	Model / Method	Qwen-0.5B	Qwen-3B	Qwen-7B	Qwen-14B	Llama-1B	Llama-3B	Llama-8B	Mistral-7B	Mistral-8B	Mistral-12B	Avg.
ArX	Log-Likelihood	0.5399	0.5882	0.5728	0.5224	0.9065	0.8997	0.8802	0.6693	0.6723	0.6696	0.6921
	Rank	0.6580	0.6585	0.6342	0.5890	0.7286	0.7178	0.6948	0.6831	0.6746	0.6904	0.6729
	Log-Rank	0.5802	0.6179	0.5989	0.5435	0.9260	0.9173	0.8944	0.6894	0.6992	0.6957	0.7163
	Entropy	0.5344	0.4906	0.4856	0.5129	0.3758	0.3588	0.3657	0.4195	0.4096	0.3858	0.4339
	LRR	0.6946	0.6943	0.6651	0.6028	0.9430	0.9308	0.8975	0.7245	0.7619	0.7579	0.7672
	Fast-DetectGPT	0.6503	0.6849	0.6523	0.5820	0.9925	0.9863	0.9736	0.7200	0.7077	0.6562	0.7606
	Binoculars	**0.8129**	**0.7543**	**0.7121**	0.6471	**0.9969**	**0.9938**	**0.9835**	0.7797	0.8400	0.7527	0.8273
	TOCSIN	0.6527	0.6866	0.6564	0.5827	0.9928	0.9865	0.9744	0.7272	0.7140	0.6632	0.7637
	TOCSIN*	0.6777	0.7106	0.6999	**0.6750**	0.9896	0.9775	0.9740	**0.8296**	**0.8918**	**0.8817**	**0.8307**
MSE	Log-Likelihood	0.4741	0.5134	0.5514	0.5171	0.7263	0.6992	0.7154	0.5850	0.5280	0.5168	0.5827
	Rank	0.5506	0.5467	0.5574	0.5471	0.6211	0.6133	0.6152	0.5837	0.5495	0.5525	0.5737
	Log-Rank	0.4841	0.5177	0.5543	0.5226	0.7265	0.7009	0.7167	0.5924	0.5299	0.5206	0.5866
	Entropy	0.5350	0.5086	0.4820	0.5013	0.3557	0.3844	0.3663	0.4763	0.5086	0.5200	0.4638
	LRR	0.5285	0.5403	0.5609	0.5440	0.6802	0.6791	0.6745	0.6183	0.5342	0.5397	0.5900
	Fast-DetectGPT	0.5426	0.6220	0.6841	0.6184	0.8925	0.8919	0.8904	0.8054	0.6986	0.6979	0.7344
	Binoculars	**0.6901**	**0.7205**	**0.7650**	**0.6817**	**0.9215**	**0.9196**	**0.9146**	**0.8685**	**0.8157**	**0.7673**	**0.8065**
	TOCSIN	0.5363	0.6162	0.6798	0.6092	0.8723	0.8765	0.8748	0.7970	0.6915	0.6908	0.7244
	TOCSIN*	0.5256	0.6371	0.6704	0.6237	0.9083	0.8979	0.9034	0.7986	0.6851	0.6893	0.7339
WP	Log-Likelihood	0.4758	0.5014	0.5324	0.4754	0.9624	0.9457	0.8346	0.6993	0.6691	0.6273	0.6723
	Rank	0.6608	0.6168	0.6097	0.5399	0.8209	0.8014	0.7139	0.6661	0.7032	0.6463	0.6779
	Log-Rank	0.5237	0.5328	0.5550	0.4931	0.9738	0.9597	0.8496	0.7142	0.6848	0.6412	0.6928
	Entropy	0.6197	0.5562	0.5204	0.5500	0.2476	0.2701	0.3453	0.4486	0.4492	0.4533	0.4460
	LRR	0.6577	0.6186	0.6068	0.5450	0.9785	0.9649	0.8477	0.7307	0.7122	0.6627	0.7325
	Fast-DetectGPT	0.6900	0.6195	0.6139	0.5508	0.9904	0.9831	0.8896	0.8222	0.7680	0.6891	0.7617
	Binoculars	0.7883	0.7070	0.7014	0.6285	0.9976	**0.9915**	0.9200	0.8581	0.8432	0.7584	0.8194
	TOCSIN	0.7072	0.6316	0.6292	0.5569	0.9892	0.9822	0.8977	0.8522	0.8109	0.7209	0.7778
	TOCSIN*	**0.7968**	**0.7863**	**0.7305**	**0.7800**	**0.9978**	0.9879	**0.9237**	**0.9432**	**0.9502**	**0.9269**	**0.8823**

Table 5: AUROC across data sources and generative models under black-box setting

Model Method	Qwen-0.5B	Qwen-3B	Qwen-7B	Qwen-14B	Avg.
Log-Likelihood	0.5292 | **0.5442**	0.5201 | **0.5666**	0.5897 | **0.6043**	0.5154 | **0.5455**	0.5386 | **0.5652**
Rank	**0.7093** | 0.5989	**0.6384** | 0.5890	**0.6480** | 0.5801	**0.6096** | 0.5734	**0.6513** | 0.5854
Log-Rank	0.5514 | **0.5558**	0.5330 | **0.5733**	0.5981 | **0.6048**	0.5277 | **0.5515**	0.5525 | **0.5714**
Entropy	**0.5316** | 0.5108	**0.5105** | 0.4731	**0.4396** | 0.4219	**0.5181** | 0.4902	**0.4999** | 0.4740
LRR	**0.6324** | 0.5992	**0.5882** | 0.5856	**0.6172** | 0.5778	**0.5841** | 0.5714	**0.6055** | 0.5835
Fast-DetectGPT	0.7455 | **0.7576**	0.6640 | **0.7257**	0.6752 | **0.6790**	0.6719 | **0.6873**	0.6891 | **0.7124**
Binoculars	**0.7073** | 0.6901	0.7158 | **0.7205**	**0.7692** | 0.7650	**0.7077** | 0.6817	**0.7250** | 0.7143
TOCSIN	0.7405 | **0.7600**	0.6507 | **0.7261**	0.6608 | **0.6769**	0.6590 | **0.6994**	0.6777 | **0.7156**
TOCSIN*	**0.7937** | 0.7602	0.7210 | **0.7351**	**0.7297** | 0.7086	0.6950 | **0.7020**	**0.7348** | 0.7265

Table 6: AUROC for MSE dataset with and without question texts (left: without question texts, right: with question texts; bold denotes maximum values in each group)

Data Source	Model Method	Qwen-0.5B	Qwen-3B	Qwen-7B	Qwen-14B	Avg.
ArX	Log-Likelihood	**0.6912** | 0.4631 | 0.5399	**0.6226** | 0.5644 | 0.5882	**0.5949** | 0.5632 | 0.5728	**0.5402** | 0.5201 | 0.5224	**0.6122** | 0.5277 | 0.5558
	Rank	**0.7060** | 0.6183 | 0.6580	0.6374 | 0.6430 | **0.6585**	0.6143 | 0.6283 | **0.6342**	0.5837 | **0.5894** | 0.5890	**0.6354** | 0.6197 | 0.6349
	Log-Rank	**0.7150** | 0.4992 | 0.5802	**0.6392** | 0.5924 | 0.6179	**0.6120** | 0.5888 | 0.5989	**0.5562** | 0.5427 | 0.5435	**0.6306** | 0.5558 | 0.5851
	Entropy	0.4004 | **0.5858** | 0.5344	0.4345 | **0.5182** | 0.4906	0.4440 | **0.5043** | 0.4856	0.4906 | **0.5273** | 0.5129	0.4424 | **0.5339** | 0.5059
	LRR	**0.7581** | 0.6232 | 0.6946	0.6691 | 0.6715 | **0.6943**	0.6447 | 0.6601 | **0.6651**	0.5960 | **0.6154** | 0.6028	**0.6670** | 0.6425 | 0.6642
	Fast-DetectGPT	**0.7229** | 0.5952 | 0.6503	0.6475 | **0.6892** | 0.6849	0.5968 | **0.6679** | 0.6523	0.5539 | **0.6089** | 0.5820	0.6303 | 0.6403 | **0.6424**
	TOCSIN	**0.7299** | 0.5936 | 0.6527	0.6532 | **0.6900** | 0.6866	0.6020 | **0.6684** | 0.6564	0.5566 | **0.6059** | 0.5827	0.6354 | 0.6395 | **0.6446**
	TOCSIN*	**0.7370** | 0.6264 | 0.6777	0.6897 | **0.7206** | 0.7106	0.6878 | **0.7220** | 0.6999	0.6556 | **0.6888** | 0.6750	**0.6925** | 0.6894 | 0.6908
MSE	Log-Likelihood	**0.5347** | 0.4431 | 0.4741	**0.5414** | 0.5017 | 0.5134	**0.5644** | 0.5447 | 0.5514	**0.5368** | 0.5092 | 0.5171	**0.5443** | 0.4997 | 0.5140
	Rank	**0.5750** | 0.5372 | 0.5506	**0.5643** | 0.5451 | 0.5467	**0.5687** | 0.5542 | 0.5574	**0.5513** | 0.5440 | 0.5471	**0.5648** | 0.5451 | 0.5505
	Log-Rank	**0.5460** | 0.4558 | 0.4841	**0.5500** | 0.5089 | 0.5177	**0.5739** | 0.5492 | 0.5543	**0.5429** | 0.5151 | 0.5226	**0.5532** | 0.5073 | 0.5197
	Entropy	0.4927 | **0.5572** | 0.5350	0.4796 | **0.5204** | 0.5086	0.4634 | **0.4897** | 0.4820	0.4845 | **0.5105** | 0.5013	0.4801 | **0.5194** | 0.5067
	LRR	**0.5753** | 0.5187 | 0.5285	**0.5726** | 0.5427 | 0.5403	**0.5950** | 0.5619 | 0.5609	**0.5558** | 0.5416 | 0.5440	**0.5747** | 0.5412 | 0.5434
	Fast-DetectGPT	**0.5915** | 0.4976 | 0.5426	0.5681 | **0.6285** | 0.6220	0.5913 | **0.6922** | 0.6841	0.5814 | **0.6259** | 0.6184	0.5831 | 0.6111 | **0.6168**
	TOCSIN	**0.5903** | 0.4944 | 0.5363	0.5654 | **0.6278** | 0.6162	0.5829 | **0.6924** | 0.6798	0.5741 | **0.6247** | 0.6092	0.5782 | 0.6098 | **0.6104**
	TOCSIN*	0.5354 | **0.5528** | 0.5256	0.5397 | **0.6464** | 0.6371	0.5886 | **0.6760** | 0.6704	0.5483 | **0.6328** | 0.6237	0.5530 | **0.6270** | 0.6142
WP	Log-Likelihood	**0.6555** | 0.3642 | 0.4758	**0.5716** | 0.4548 | 0.5014	**0.5809** | 0.4922 | 0.5324	**0.5075** | 0.4480 | 0.4754	**0.5789** | 0.4398 | 0.4962
	Rank	**0.6616** | 0.6150 | 0.6608	0.5802 | 0.6063 | **0.6168**	0.5617 | 0.6033 | **0.6097**	0.5155 | 0.5313 | **0.5399**	0.5797 | 0.5890 | **0.6068**
	Log-Rank	**0.7120** | 0.4014 | 0.5237	**0.6101** | 0.4802 | 0.5328	**0.6133** | 0.5120 | 0.5550	**0.5378** | 0.4607 | 0.4931	**0.6183** | 0.4636 | 0.5261
	Entropy	0.4726 | **0.6873** | 0.6197	0.4872 | **0.5994** | 0.5562	0.4554 | **0.5642** | 0.5204	0.4991 | **0.5793** | 0.5500	0.4786 | **0.6075** | 0.5616
	LRR	**0.7989** | 0.5584 | 0.6577	**0.6815** | 0.5782 | 0.6186	**0.6663** | 0.5831 | 0.6068	**0.6043** | 0.5152 | 0.5450	**0.6878** | 0.5587 | 0.6070
	Fast-DetectGPT	**0.6921** | 0.6068 | 0.6900	0.5868 | 0.6157 | **0.6195**	0.5703 | **0.6261** | 0.6139	0.5113 | **0.5596** | 0.5508	0.5901 | 0.6020 | **0.6186**
	TOCSIN	**0.7209** | 0.6133 | 0.7072	0.6269 | 0.6216 | **0.6316**	0.6111 | **0.6357** | 0.6292	0.5554 | **0.5635** | 0.5569	0.6286 | 0.6085 | **0.6312**
	TOCSIN*	0.7920 | 0.7563 | **0.7968**	0.7865 | **0.7942** | 0.7863	**0.7535** | 0.7493 | 0.7305	**0.8272** | 0.7970 | 0.7800	**0.7898** | 0.7742 | 0.7734

Table 7: AUROC with different proxy models under black-box setting (left to right: GPT-2 (Radford et al., 2019), GPT-J (Wang and Komatsuzaki, 2021), NEO-2.7 (Black et al., 2021); boldface and underlined text denote the maximum and second-maximum values, respectively, within each group)