Complete Evasion, Zero Modification: PDF Attacks on AI Text Detection

Aldan Creo
Independent Author
Dublin, Ireland
research@acmc.fyi

Abstract

AI-generated text detectors have become essential tools for maintaining content authenticity, yet their robustness against evasion attacks remains questionable. We present PDFuzz, a novel attack that exploits the discrepancy between visual text layout and extraction order in PDF documents. Our method preserves exact textual content while manipulating character positioning to scramble extraction sequences. We evaluate this approach against the ArguGPT detector using a dataset of human and AI-generated text. Our results demonstrate complete evasion: detector performance drops from $(93.6 \pm 1.4)\,\%$ accuracy and 0.938 ± 0.014 F1 score to random-level performance ($(50.4 \pm 3.2)\,\%$ accuracy, 0.0 F1 score) while maintaining perfect visual fidelity. Our work reveals a vulnerability in current detection systems that is inherent to PDF document structures and underscores the need for implementing sturdy safeguards against such attacks. We make our code publicly available at https://github.com/ACMCMC/PDFuzz.

1 Introduction

The proliferation of large language models has created an urgent need for reliable AI-generated text detection systems. These detectors serve critical functions in academic integrity, content verification, and misinformation prevention. However, recent research has demonstrated that many detection systems can be circumvented through various evasion techniques.

Existing evasion methods typically involve content modification: paraphrasing attacks alter semantic meaning (Krishna et al., 2023), character substitution techniques replace visually similar characters (Creo and Pudasaini, 2025), and adversarial perturbations introduce subtle textual changes (Huang et al., 2024b). While effective, these approaches either modify the original content or introduce visually detectable artifacts. Table 1 summarizes the key characteristics of existing evasion attacks and their limitations.

We identify a previously unexplored vulnerability in PDF document format: the distinction between visual character layout and text extraction order. PDF viewers display characters based on their spatial coordinates, yet text extraction follows the order in which characters are written to the document. This discrepancy creates an opportunity for evasion attacks that preserve exact textual content while scrambling extraction sequences.

Our contribution is threefold: (1) we introduce PDFuzz, the first PDF-based text ordering attack against AI detectors, (2) we demonstrate complete evasion against state-of-the-art detection systems while maintaining perfect visual fidelity, and (3) we provide empirical evidence of fundamental vulnerabilities in current detection methodologies. PDFuzz reduces the ArguGPT detector from 93.6 % accuracy to essentially random performance without modifying a single character.

2 Methods

2.1 PDF Text Representation

Unlike structured markup languages such as HTML or XML, PDF documents represent text as sequences of low-level printing commands rather than logical document structures (Livathinos et al., 2021). Each character is positioned using absolute coordinates within a page coordinate system, where the origin (0,0) typically corresponds to the bottom-left corner of the page, with x-coordinates increasing rightward and y-coordinates increasing upward.

PDF text positioning relies on specific operators that control both character content and spatial placement (ISO, 2008). The fundamental text-showing operators include:

- `Tj` for displaying a string at the current text position

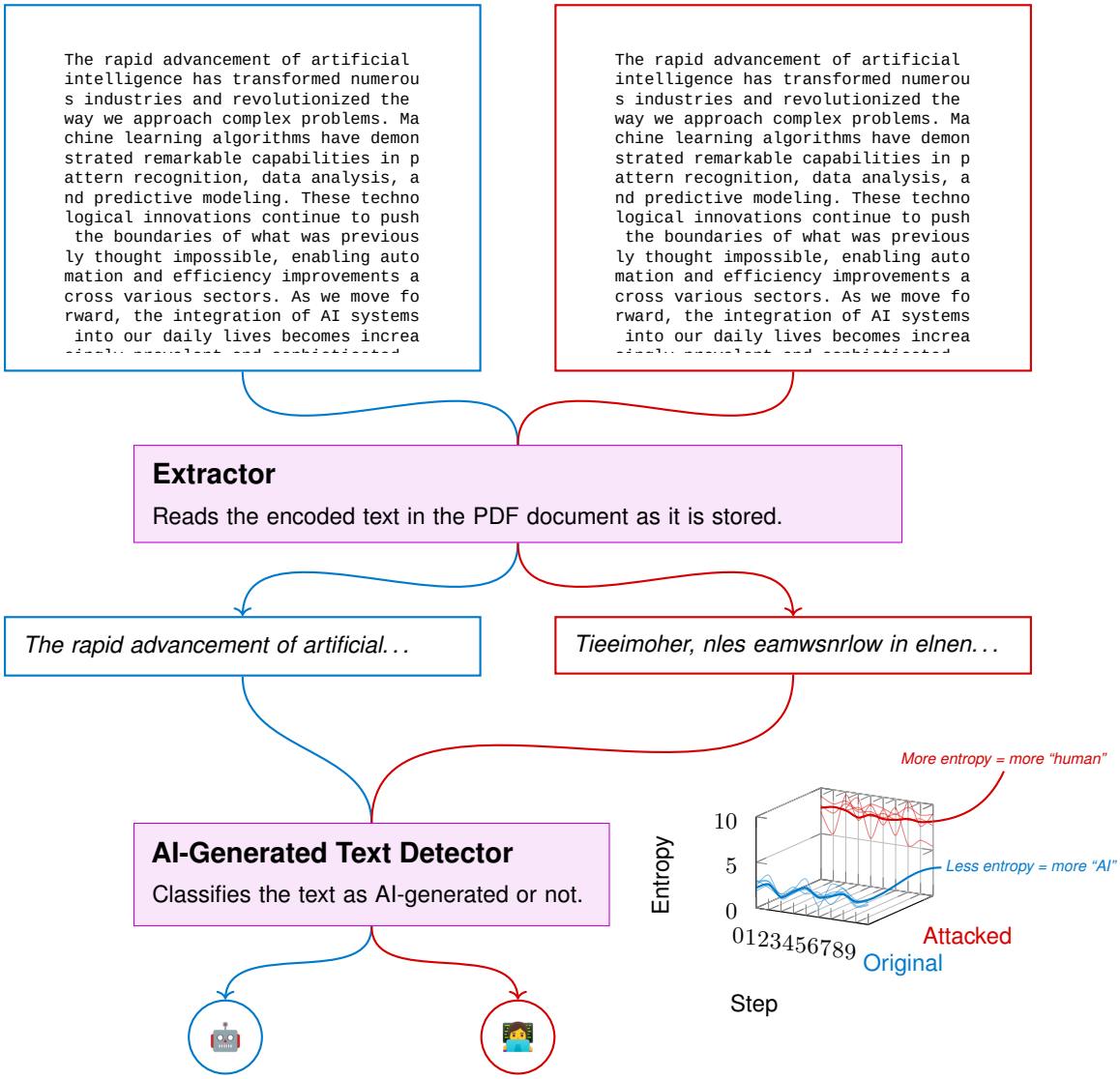

Figure 1: **PDFuzz.** This figure illustrates our attack methodology (as a simplified example): (1) a **Normal PDF** contains AI-generated text with standard character ordering, appearing identical to (2) an **Attacked PDF** which uses scrambled character positioning while maintaining visual fidelity (select the texts to inspect the differences). (3) To perform AI detection, an **extractor** needs to process both documents according to their internal structure. (4) Extraction yields coherent text from the normal PDF but scrambled sequences from the attacked version. (5) The **AI-generated text detector** correctly identifies the normal text as AI-generated (🤖) but misclassifies the scrambled text as human-written (👩‍💻). The 3D plot shows perplexity patterns: original text maintains low, stable perplexity while attacked text exhibits high, erratic perplexity due to character scrambling, causing detection failure.

Attack Method	Same Content	Same Appearance	Same Meaning	Main Claims
DIPPER Paraphrasing (Krishna et al., 2023)	✗	✗	✗	Their biggest drop in detection accuracy (at 1% FPR) is from 70.3 % to 4.6 %; smallest from 100.0 % to 55.8 %.
Adversarial Paraphrasing (Cheng et al., 2025)	✗	✗	✗	Average T@1%F reduction of 80.75 % to 87.88 %, depending on the detector.
Word/Sentence Substitution (Peng et al., 2023)	✗	✗	✗	Their perturbation methods generally reduce detection accuracy to 50-60%.
Token-Level Blending (Huang et al., 2024a)	✗	✗	✗	Best result is an AUROC reduction from 0.9845 to 0.3968, with most configurations showing modest results and some remaining unaffected.
Homoglyph Substitution (Creo and Pudasaini, 2025)	✗	∼	✓	Decrease the average Matthews Correlation Coefficient from 0.64 to −0.01.
Adversarial Perturbations (Zhou et al., 2024)	✗	∼	∼	AUC dropped from 99.63 % to 51.06 %.
PDFuzz (Ours)	✓	✓	✓	**Complete evasion (F1 → 0.0)**

Table 1: Comparison of evasion attacks on AI text detectors. ✗ indicates modification/change, ∼ indicates minimal change, ✓ indicates no modification/change. PDFuzz is the only method that achieves complete evasion without any content, visual, or semantic modifications.

Note: We include main claims rather than numerical metrics because results are not directly comparable across different datasets, evaluation metrics, and experimental conditions. For complete details, we refer readers to the original articles.

`TJ` for displaying an array of strings and numeric adjustments that allow fine-tuned character spacing

`Tm` for setting the text matrix that defines character positioning and scaling

`Td` for moving to a new position relative to the current location

These operators enable precise control over character placement independent of reading order.

The PDF coordinate system thus allows characters to be placed at any location of choice. While conventional PDF generation writes characters sequentially from left to right and top to bottom, the specification imposes no constraints on the order in which positioning commands appear in the document stream. This flexibility enables our attack: characters can be written to the PDF in any sequence while maintaining their visual positions through explicit coordinate specification.

Text extraction tools process PDF documents by parsing the sequence of positioning commands as they appear in the document stream (Zhu and Cole, 2022). Extraction order thus follows the order of text-showing operators in the PDF file, not the visual left-to-right, top-to-bottom reading order. This is the discrepancy that we exploit to create a gap between visual presentation and extraction sequence. In other words, while a PDF viewer makes it appear that characters are arranged in a natural reading order, we manipulate the underlying structure so that automated readers extract the text in a scrambled manner.

2.2 Threat Model

We assume an attacker with the following capabilities: (1) access to AI-generated text that requires evasion from detection systems and (2) ability to convert text into PDF format before submission to detectors. This represents a typical setup in most contexts of daily life, such as students submitting essays or researchers sharing manuscripts. The attacker cannot modify the original text content and must preserve visual fidelity to avoid human detection.

We assume detectors operate on extracted text sequences without format-aware preprocessing. This reflects current practice where detection systems typically receive plain text input after document parsing, making them vulnerable to extraction-order manipulation. The attack's effectiveness depends on the detection system processing text in the order defined by the PDF file structure, which should be the case for readers compliant with the PDF specification (ISO, 2008). We do not assume access to detector internals or training data.

Our attack operates under strict constraints: no modification of character content, preservation of exact visual layout, and maintenance of document

readability for human users. These constraints distinguish our approach from existing evasion methods that alter semantic content or introduce visual artifacts.

2.3 Attack Methodology

Our attack methodology operates in two phases. First, we analyze the target text to determine optimal character positions for normal visual presentation using monospace fonts to ensure consistent spacing. Second, we generate a modified PDF where characters are written in scrambled order while maintaining identical spatial positioning.

We implement two scrambling strategies: character-level randomization and chunk-based reordering. Character-level scrambling randomly permutes individual character positions throughout the document. Chunk-based reordering divides the text into segments of 8-15 characters and shuffles these segments while preserving intra-chunk order. Both methods preserve exact character content and visual layout.

Our implementation uses the ReportLab library to generate PDFs with precise character positioning. We calculate character coordinates based on monospace font metrics and page margins. Currently, we employ monospace fonts to simplify character alignment calculations, as each character occupies identical horizontal space. While this approach ensures reliable positioning, future implementations could incorporate sophisticated typesetting engines to support proportional fonts with varying per-glyph widths. Such enhancement would require computing individual character metrics and adjusting positions accordingly, but we leave this optimization for future work as our focus remains on demonstrating the core vulnerability rather than optimizing visual presentation.

The scrambling process maintains one-to-one character correspondence between normal and attacked versions, ensuring identical visual presentation and character counts. We verify attack success through both automated metrics comparing extracted text sequences and manual visual inspection of rendered PDFs.

3 Results

We evaluate our attack against the ArguGPT detector (Liu et al., 2023), a RoBERTa-large model fine-tuned for identifying AI-generated argumentative essays. Our evaluation uses 1,000 essays from the SilverSpeak dataset, comprising both human-written and AI-generated content. We generate both normal PDFs and attacked versions using character-level scrambling.

The ArguGPT detector demonstrates strong baseline performance on normal text extraction, achieving $(93.6 \pm 1.4)\,\%$[1] accuracy and 0.938 ± 0.014 F1 score. The detector maintains high precision at low false positive rates, with 73.4 % true positive rate at 1 % false positive rate.

Our PDFuzz attack reduces detector performance to random levels. Accuracy drops to $(50.4 \pm 3.2)\,\%$ and F1 score collapses to 0.0. The true positive rate at 1 % false positive rate falls to 0.2 %, indicating complete loss of discriminative capability.

Manual inspection of generated PDFs can confirm the identical visual presentation between normal and attacked versions (Figure 1).

4 Discussion

The attack's effectiveness stems from AI detectors' reliance on sequential text processing. Models like ArguGPT expect text to follow natural reading order, with statistical patterns reflecting normal language generation. When character extraction order becomes scrambled, these patterns are disrupted, causing detectors to misclassify the content.

This vulnerability is fundamental because it exploits the gap between human visual perception and machine text processing. While humans process text through visual pattern recognition, AI detectors analyze extracted character sequences. This creates an attack surface that preserves human-readable content while confusing machine analysis.

Potential defenses include text normalization before detection and PDF structure analysis to identify manipulation. However, these approaches face practical limitations: normalization might alter legitimate formatting, while structure analysis could be circumvented through more sophisticated positioning techniques.

5 Conclusion

We present PDFuzz, the first PDF-based text ordering attack against AI-generated text detectors. Our method achieves complete evasion without modifying textual content, revealing fundamental vulnerabilities in current detection systems. PDFuzz

[1] 95% confidence interval.

demonstrates that effective evasion need not compromise visual fidelity or semantic content.

These findings highlight the need for more robust evaluation methodologies that consider document format vulnerabilities. Future detection systems should account for the distinction between visual presentation and underlying data structure across different document formats.

Limitations

Our attack is specific to PDF documents and may not generalize to other formats. The effectiveness depends on the target detector's reliance on character-level sequential processing. Additionally, sophisticated detectors might implement format-aware preprocessing that could mitigate this vulnerability.

Ethics Statement

Our research aims to strengthen AI detection systems by identifying vulnerabilities. We do not intend to exacerbate negative AI-related effects, such as academic misconduct or disinformation. In fact, our goal is to raise awareness of a significant vulnerability affecting current detectors. Similarly, we are making our code and techniques publicly available for academic research purposes only, and we do not allow their use for any other purpose without explicit prior authorization.

We hope our work will contribute to the development of robust detection systems which are used in a responsible manner for the greater good of society.

References

Yize Cheng, Vinu Sankar Sadasivan, Mehrdad Saberi, Shoumik Saha, and Soheil Feizi. 2025. Adversarial paraphrasing: A universal attack for humanizing ai-generated text. *Preprint*, arXiv:2506.07001.

Aldan Creo and Shushanta Pudasaini. 2025. SilverSpeak: Evading AI-generated text detectors using homoglyphs. In *Proceedings of the 1stWorkshop on GenAI Content Detection (GenAIDetect)*, pages 1–46, Abu Dhabi, UAE. International Conference on Computational Linguistics.

Fan Huang, Haewoon Kwak, and Jisun An. 2024a. To-blend: Token-level blending with an ensemble of llms to attack ai-generated text detection. *arXiv preprint arXiv:2402.04776*.

Guanhua Huang, Yuchen Zhang, Zhe Li, Yongjian You, Mingze Wang, and Zhouwang Yang. 2024b. Are AI-generated text detectors robust to adversarial perturbations? In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6005–6024, Bangkok, Thailand. Association for Computational Linguistics.

ISO. 2008. Document management – portable document format – part 1: Pdf 1.7. Technical Report ISO 32000-1:2008, International Organization for Standardization. Available from Adobe at `https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/PDF32000_2008.pdf`.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Frederick Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. In *Thirty-seventh Conference on Neural Information Processing Systems*.

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

Nikolaos Livathinos, Cesar Berrospi, Maksym Lysak, Viktor Kuropiatnyk, Ahmed Nassar, Andre Carvalho, Michele Dolfi, Christoph Auer, Karel Dinkla, and Peter Staar. 2021. Robust pdf document conversion using recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 15137–15145.

Xinlin Peng, Ying Zhou, Ben He, Le Sun, and Yingfei Sun. 2023. Hidding the ghostwriters: An adversarial evaluation of AI-generated student essay detection. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Ying Zhou, Ben He, and Le Sun. 2024. Humanizing machine-generated content: Evading AI-text detection through adversarial attack. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 8427–8437, Torino, Italia. ELRA and ICCL.

Miao Zhu and Jacqueline Cole. 2022. Pdfdataextractor: A tool for reading scientific text and interpreting metadata from the typeset literature in the portable document format. *Journal of Chemical Information and Modeling*, 62(7):1633–1643.