Computers, Materials & Continua

Tech Science Press

Doi:10.32604/cmc.2025.073347

REVIEW

AI-Generated Text Detection: A Comprehensive Review of Active and Passive Approaches

Lingyun Xiang[1,*], **Nian Li**[2], **Yuling Liu**[3] **and Jiayong Hu**[1]

[1]School of Computer Science and Technology, Changsha University of Science and Technology, Changsha, 410076, China
[2]School of Physics and Electronic Science, Changsha University of Science and Technology, Changsha, 410076, China
[3]College of Cyber Science and Technology, Hunan University, Changsha, 410082, China

*Corresponding Author: Lingyun Xiang. Email: xiangly210@163.com

Received: 16 September 2025; Accepted: 20 November 2025; Published: 12 January 2026

ABSTRACT: The rapid advancement of large language models (LLMs) has driven the pervasive adoption of AI-generated content (AIGC), while also raising concerns about misinformation, academic misconduct, biased or harmful content, and other risks. Detecting AI-generated text has thus become essential to safeguard the authenticity and reliability of digital information. This survey reviews recent progress in detection methods, categorizing approaches into passive and active categories based on their reliance on intrinsic textual features or embedded signals. Passive detection is further divided into surface linguistic feature-based and language model-based methods, whereas active detection encompasses watermarking-based and semantic retrieval-based approaches. This taxonomy enables systematic comparison of methodological differences in model dependency, applicability, and robustness. A key challenge for AI-generated text detection is that existing detectors are highly vulnerable to adversarial attacks, particularly paraphrasing, which substantially compromises their effectiveness. Addressing this gap highlights the need for future research on enhancing robustness and cross-domain generalization. By synthesizing current advances and limitations, this survey provides a structured reference for the field and outlines pathways toward more reliable and scalable detection solutions.

KEYWORDS: AI-generated text detection; large language models; text classification; watermarking

1 Introduction

In recent years, the rapid development of large language models (LLMs) such as the GPT series [1], BERT [2], T5 [3], and the recently open-sourced DeepSeek [4], has significantly propelled the progress of natural language generation (NLG). These models have demonstrated outstanding performance in tasks such as text summarization, machine translation, and content creation, producing outputs that are often stylistically and semantically indistinguishable from human writing. However, the increasing sophistication and accessibility of LLMs have introduced numerous challenges, including intellectual property infringement, the spread of misinformation, biased or discriminatory content, privacy breaches, malicious exploitation, academic plagiarism, fabricated news, and fake reviews on social media platforms [5–7]. While LLMs can generate coherent and fluent text, they remain susceptible to producing unreliable or fabricated content. As AI-generated text becomes increasingly ubiquitous online, the need for reliable detection mechanisms has become urgent and indispensable. In response, researchers have focused on AI-generated text detection (AIGTD) as a safeguard for information authenticity, driving the evolution of detection techniques from early statistical heuristics to the modern approaches explored today.

Early research on AI-generated text detection primarily employed statistical heuristics or basic classification models. The emergence of ChatGPT marked a watershed moment, drastically expanding both the capabilities of LLMs and the challenges of detecting their outputs. Since then, research attention has shifted toward more complex scenarios, such as distinguishing AI-generated text from human-authored content, evaluating the robustness of detection systems, and developing defenses against adversarial attacks. Typically, AIGTD is formulated as a binary classification problem, as illustrated in Fig. 1. Given an input text sample $s \in S$, the detector assigns a label $y \in \{0, 1\}$, where $y = 1$ denotes AI-generated text and $y = 0$ denotes human-authored text. The detector $D(s)$ can be a probabilistic model or a discriminative classifier designed to infer the origin of the input text.

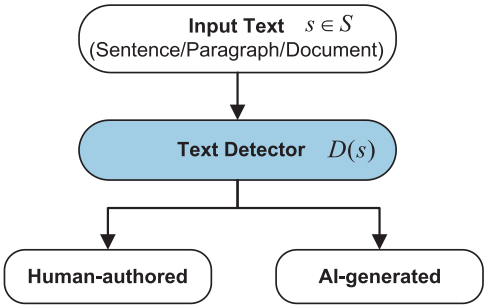

Figure 1: Binary classification framework for AI-generated text detection

Current detection methods can be broadly categorized into two paradigms: passive and active detection. Passive detection does not require access to the text generation process; it discriminates between human-authored and AI-generated text by analyzing linguistic features, statistical patterns, or semantic coherence, among other factors. This approach offers high flexibility and ease of deployment; however, it often exhibits limited generalization across domains and remains vulnerable to evasion through paraphrasing or style manipulation. In contrast, active detection embeds identifiable signals—such as watermarks—into the text during or after generation to enable provenance tracking. While generally more robust and tamper-resistant, this approach presents substantial challenges for integration and deployment in real-world settings.

Building on this categorization, this paper presents a systematic review of AI-generated text detection. Unlike prior surveys that often emphasize either watermarking or linguistic analysis in isolation, our work provides a holistic taxonomy that jointly considers both passive and active paradigms. By integrating recent advances into this framework, we aim to clarify the methodological landscape, highlight the connections and distinctions among different detection strategies, and offer readers a structured reference for navigating this rapidly evolving field. On this basis, this review seeks not only to consolidate fragmented research efforts but also to provide a foundation for future investigations into reliable and scalable detection solutions.

The remainder of this review is organized as follows. Section 2 provides an overview of AIGTD, including task definitions, challenges, and a taxonomy of existing methods. Section 3 discusses passive detection methods, which infer authorship based solely on the properties, including surface-linguistic feature-based and language model-based approaches. Section 4 focuses on active detection approaches, covering watermarking-based and retrieval-based methods. Section 5 highlights current limitations and emerging challenges, outlining potential research directions. Finally, Section 6 concludes the paper by summarizing key insights and implications for future work.

2 Overview of AI-Generated Text Detection

This section provides a structured overview of AIGTD. It begins with the general workflow of text classification and its distinctions from AIGTD. It then examines the text generation mechanisms of LLMs, emphasizing the linguistic and statistical properties that are particularly relevant to distinguishing AI-generated text from human-authored text. The section further considers the principal challenges that complicate reliable detection at the technical, model, and application levels, followed by a taxonomy that organizes existing detection methods into passive and active paradigms to clarify the methodological landscape.

2.1 Task Formulation and Foundations

As outlined in Section 1, AIGTD is typically formulated as a binary classification problem, where the objective is to distinguish between human-authored and AI-generated text. While this binary framing captures the essence of the task, practical settings often extend beyond it. For instance, detection may be designed as a multi-class problem to identify outputs from different LLMs, or as an open-set problem to handle previously unseen generative systems. The theoretical foundation of this task lies in text classification [8], which is a fundamental task in NLP. Text classification aims to automatically assign a text sample to one or more predefined categories. It includes binary, multi-class, and multi-label classification, and is widely used in applications such as sentiment analysis, spam detection, and topic labeling. A typical pipeline involves three steps:

(1) **Text representation:** transforming raw text into structured features using techniques such as bag-of-words, TF-IDF, or word embeddings;
(2) **Classifier modeling:** building classifiers with statistical methods (e.g., SVM, Naive Bayes) or deep neural architectures (e.g., CNN, RNN, Transformer);
(3) **Decision output:** producing a probability distribution over the discrete class labels and determining the final predicted category.

Although AIGTD inherits the general pipeline of text classification, it differs substantially in objectives, feature dependencies, and operational complexity. Traditional classification tasks, such as sentiment analysis or topic labeling, are predominantly content-driven, focusing on semantic signals directly aligned with the label. Even when employing deep encoders, these models learn meaning-oriented representations tied to topics, stances, or sentiments. In contrast, AIGTD is largely authorship-driven or process-driven, relying on linguistic and generative regularities that are weakly coupled to content. Such cues include lexical choice distributions and burstiness, syntactic and stylistic regularities, discourse-level coherence, and probability calibration or perplexity profiles. In addition, AIGTD detectors must operate in dynamic and adversarial environments shaped by rapidly evolving generative models and deliberate attempts to evade detection. Improvements in language models have significantly narrowed surface-level distinctions (e.g., n-gram frequency, sentence length) that traditional text classifiers exploit. Meanwhile, detectors trained on structured domains such as news or academic writing often struggle to generalize effectively to informal text, including dialogue and social media [9]. The rapid evolution of generative models (e.g., from GPT-3 to GPT-5) also introduces frequent distribution shifts, necessitating continuous updates to detectors to maintain effectiveness. These differences are summarized in Table 1.

Table 1: Key differences between traditional text classification and AI-generated text detection methods

	Traditional text classification	AI-generated text detection
Task objective	Multi-class classification (e.g., sentiment analysis, topic classification)	Binary discrimination (Human vs. AI)
Feature reliance	Content-driven semantic representations	Authorship/process-driven linguistic and generative regularities
Data environment	Relatively static	Evolving distributions; adversarial obfuscation (e.g., paraphrasing, style transfer)
Domain scope	Domain-specific (e.g., news, reviews)	Cross-domain (e.g., dialogue, summarization, academic writing)
Classification boundary	Clear and well-separated	Blurred and dynamic

In summary, while AIGTD can be formally situated within the paradigm of text classification, it requires more advanced semantic reasoning and modeling techniques to capture subtle inconsistencies, adapt to shifting generative patterns, and ensure robustness across diverse domains.

2.2 Text Generation Mechanisms of Large Language Models

Detecting AI-generated text requires a deep understanding of how large language models (LLMs) produce text. LLMs are trained on massive corpora to learn the conditional distributions of tokens, enabling them to model and generate texts that are syntactically coherent and semantically plausible. Formally, the training objective of an LLM is to maximize the likelihood of observing a token sequence by predicting the next token given its preceding context:

$$L_{LM}(x) = \sum \log P(x_t | x_{<t}; \theta) \tag{1}$$

where x_t denotes the current t-th predicted token, $x_{<t}$ represents the historical context, and θ refers to the model parameters. This autoregressive approach allows the model to recursively generate text, with each selected token appended to the sequence for subsequent prediction.

During generation, the output is shaped not only by the learned conditional distribution but also by decoding strategies that trade off determinism and diversity. Common approaches include greedy decoding, temperature sampling, top-k sampling, and nucleus sampling [10]. These techniques directly affect lexical choice, stylistic variability, and the overall naturalness of generated text.

As model scale continues to expand, LLMs demonstrate a set of "emergent abilities", such as in-context learning, instruction following, and chain-of-thought reasoning [11]. These capabilities substantially enhance the coherence, informativeness, and stylistic sophistication of generated outputs. To further align generations with human intent, LLMs are commonly refined through instruction tuning and reinforcement learning with human feedback (RLHF) [12]. Instruction tuning adapts models to standardized prompts and task specifications, while RLHF incorporates human preference modeling and strategy optimization to ensure generated content aligns with human preferences, ethical norms, and usage scenario requirements.

Despite these advances, LLM-generated text remains a probabilistic sampling process subject to limitations. Biases in training data, stochasticity in sampling strategies, and context window constraints may introduce artifacts such as "hallucinations" or logical inconsistencies [13,14]. Although often subtle, these imperfections, together with statistical regularities in lexical distributions, stylistic uniformity, and discourse-level structures, constitute the very signals that current AIGTD methods seek to identify and exploit, thereby underpinning many existing detection approaches.

2.3 Challenges in the AI-Generated Text Detection Task

The improved realism and diversity of LLM-generated outputs have made the task of detecting AI-generated text increasingly challenging. Unlike traditional text classification tasks, AIGTD confronts multifaceted obstacles. These include the inherent difficulty of distinguishing between human and machine outputs, limitations imposed by access constraints to black-box models, and uncertainties regarding the generalizability of models across diverse domains and languages. A recent empirical study by Nabata et al. [15] found that human experts not only failed to distinguish AI-generated scientific abstracts from originals reliably but also demonstrated a significant preference for the AI-generated versions, citing their superior structure. This highlights the unreliability of manual detection and the challenge posed by the high quality of machine-generated text. In summary, AIGTD involves three significant types of complexity: technical mechanisms, model environments, and application contexts. Each of these aspects is characterized by a high degree of uncertainty.

(1) Technical Level: The linguistic similarity between AI-generated and human-authored text has increased substantially with the advent of advanced models such as GPT-4 [16]. These models can produce outputs that match or even surpass human writing in terms of perplexity, syntactic complexity, and information density, thereby eroding traditional discriminative features. Moreover, the adversarial robustness of AI-generated text detectors remains fragile: even simple paraphrasing techniques, such as synonym substitution or sentence reordering, have been shown to reduce the accuracy of systems like DetectGPT from nearly 70% to below 5% [17]. Another bottleneck is text length. In short-text scenarios, most detectors typically exhibit a marked decline in performance compared with long-text settings [18].

(2) Model Level: Detection often occurs in black-box settings, especially when dealing with proprietary APIs like GPT-4 Turbo. These systems expose neither logits nor token probabilities, rendering zero-shot detection methods relying on probability curvature analysis (e.g., DetectGPT [19]) infeasible. In addition, robust cross-model detection is challenged by the diversity of generative systems. Models such as GPT, LLaMA, and Claude differ in architectures, stylistic tendencies, and output distributions, creating substantial barriers to generalization across heterogeneous model families.

(3) Application Level: As AI-generated text becomes increasingly prevalent in various fields such as education, news, and social media, detection tasks are gradually expanding toward multilingual and multi-scenario applications. Cross-language detection necessitates adaptability to diverse morphological structures and discourse styles, thereby further complicating robustness and generalization. In practical deployments, AI-generated text often appears in hybrid forms, such as mixed human-machine writing, lightly edited content, or iteratively revised drafts, further blurring the boundaries of text origins and complicating classification. Beyond technical considerations, AIGTD intersects with broader societal issues, including academic integrity, content responsibility, and legal boundaries. Especially in high-sensitivity scenarios, misclassifying human-authored text as AI-generated entails serious ethical, reputational, and legal consequences. Consequently, improving the accuracy and fairness of detection systems has become a key research direction.

2.4 Taxonomy of AI-Generated Text Detection Methods

In response to the technical and practical challenges outlined above, numerous detection methods have been developed in recent years. These methods differ in terms of how they access information, the linguistic or statistical cues they exploit, and their resilience against evasion strategies. To provide conceptual clarity, existing AIGTD methods can be broadly categorized by whether they depend on embedded signals or inherent textual features. This yields two major detection paradigms: passive detection, which relies solely on analyzing the generated text, and active detection, which requires embedding auxiliary signals into the generation process. Each paradigm can be further divided into distinct technical pathways, as illustrated in Fig. 2.

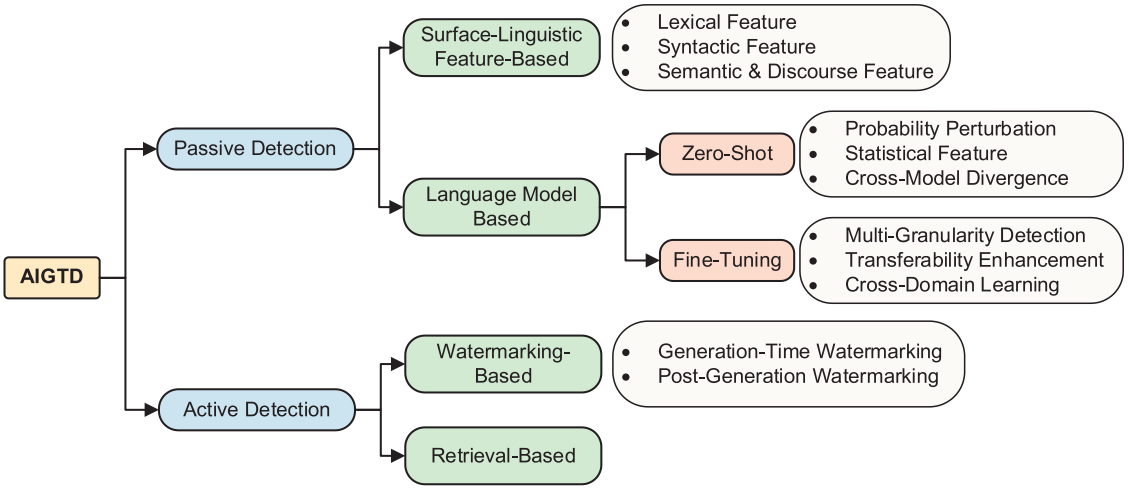

Figure 2: Taxonomy of active and passive AI-generated text detection approaches

Passive detection. Passive methods determine whether a text is AI-generated by analyzing its linguistic characteristics or statistical regularities without relying on the text generation process or embedding additional information. They are particularly suited for detecting content from unknown or untrusted sources, offering strong versatility and broad applicability. Passive detection methods can be broadly grouped into two categories:

(1) Surface linguistic feature-based methods, which examine shallow linguistic and statistical cues across multiple dimensions, including lexical, syntactic, semantic, and discourse features, such as vocabulary richness, sentence structure, semantic coherence, *n*-gram frequency, and perplexity. These features often exhibit distinctive patterns between human-authored and AI-generated text, providing informative signals for detection. However, their discriminative power tends to diminish as generative models advance, closely mimicking human writing.

(2) Language model-based methods use large language models to score text under learned distributions and include two lines: zero-shot and fine-tuned detection. Zero-shot methods require no labeled data and probe the model's probability space: via perturbation tests, perplexity/token-likelihood features, or cross-model divergence, making them lightweight and easy to deploy in low-resource settings. By contrast, fine-tuned detectors adapt pre-trained models on labeled human- and AI-generated corpora to learn discriminative distributional patterns, including work on multi-granularity features, transferability across generators, and cross-domain generalization. While fine-tuned models often achieve higher in-distribution accuracy, robustness to paraphrasing and domain shift remains a challenge.

Active detection. Active detection methods embed identifiable signals into text either during the generation process or after the text has been produced. These signals serve as embedded markers, such as watermarks or semantic signatures, that facilitate provenance verification. Among active methods, watermarking has been the most widely studied and can be divided into two types:

(1) Generation-time watermarking, which modifies the token-sampling distribution (e.g., logit biasing, vocabulary partition, entropy-gated selection) so that generated text carries statistically testable traces. Thresholds are calibrated to balance between reliability and false-positive rates.
(2) Post-generation watermarking, which embeds a short binary identifier by applying semantically constrained synonym substitution or controlled paraphrasing while preserving meaning.

In addition to watermarking, another active strategy is retrieval-based detection, which encodes the detected text into a semantic vector space and compares it against a database of previously generated outputs. If the maximum similarity score exceeds a predefined threshold, the text is recognized as AI-generated. Watermarking generally supports single-text verification but degrades under heavy paraphrasing or lossy transformations. Retrieval-based methods are comparatively robust to light edits when comprehensive generation logs are available, yet they cannot handle unlogged outputs and may raise privacy concerns.

3 Passive Detection Methods

Passive detection methods analyze the linguistic and statistical properties of text to distinguish between human-authored and AI-generated text. Building on the taxonomy outlined in Section 2.4, this section reviews their main methodological directions, focusing on surface linguistic feature-based detection (Section 3.1) and language model-based detection (Section 3.2).

3.1 Surface Linguistic Feature-Based Detection

Surface linguistic feature-based detection methods constitute one of the earliest and most widely explored approaches in AIGTD. As illustrated in Fig. 3, these methods follow a general workflow: undergo preprocessing steps such as tokenization, part-of-speech tagging, or function word tagging, after which features are extracted across multiple linguistic levels, including lexical, syntactic, and semantic/discourse attributes. These features are then subjected to feature selection and fusion, ensuring that the most discriminative indicators are retained and combined into a unified representation. The resulting feature vectors are fed into downstream classifiers to determine whether the text is human- or AI-generated. Classifiers range from traditional machine learning algorithms (e.g., SVM, Random Forests, Naive Bayes, Decision Trees) to neural architectures (e.g., CNNs, RNNs, LSTMs). More recent work has explored hybrid frameworks that integrate engineered linguistic indicators with deep learning, thereby enhancing robustness and cross-domain generalization.

Research in this line has investigated a wide range of features across multiple linguistic dimensions:

(1) Lexical features primarily focus on metrics such as word frequency distributions, lexical diversity, and repetition patterns. Research has shown that AI-generated text exhibits a significant tendency towards using structured and repetitive vocabulary. Fröhling and Zubiaga [20] proposed a linguistic feature ensemble method, noting that text from automated language models often displays fixed distribution patterns in common words, named entities, and stop words, resulting in significantly lower lexical diversity than human-authored text. Guo et al. [21] introduced syntactic-discourse profiling. They observed that language models, compared with human authors, have a higher propensity to use nouns, determiners, and conjunctions, leading to a more objective and structured writing style. In the context

of Chinese, Wang et al. [22] developed a linguistic-statistical classifier ensemble. They found that AI-generated abstracts displayed substantial homogeneity and logical regularity, while human-written abstracts reflected greater individual variation and policy-specific terminology. Additionally, Gallé et al. [23] proposed an unsupervised distributional detection method, further highlighting the high degree of repetition in AIGT, particularly in higher-order *n*-grams.

(2) Syntactic features typically capture sentence-level structural differences in sentence length, syntactic complexity, and logical organization, offering valuable cues for detection at the syntactic level. Uchendu et al. [24] introduced a neural authorship attribution framework that analyzes AI-generated text using Flesch reading scores and syntactic complexity metrics. They found that AI-generated text often exhibits monotonous sentence structures, unclear logical hierarchies, and a lack of diverse rhetorical devices. Fröhling and Zubiaga [20] corroborated these findings by leveraging part-of-speech distributions and syntactic depth as input features to train multiple classifiers. In a Chinese context, Wang et al. [22] also reported that AI-generated academic abstracts tend to use longer and more structurally consistent sentences, whereas human texts employ more varied sentence breaks and emphases. These differences provide a crucial entry point for detection at the syntactic level.

(3) Semantic and discourse features emphasize coherence, logical consistency, and connectivity across sentences and paragraphs. Liu et al. [25] introduced a novel coherence representation model COCO, which constructs entity coherence graphs to model inter-sentential semantic relations, achieving strong performance even in low-resource settings. Hamed et al. [26] proposed the xFakeSci system, which is based on the statistical similarity of bigrams, and showed that ChatGPT-generated scientific text covers only 23% of academic bigrams. By comparing TF-IDF scores, they further found that bigrams generated by ChatGPT were significantly deficient in technical terminology, highlighting a substantial terminological gap between human authors and LLM-generated content. At a deeper level of discourse organization, Kim et al. [27] proposed the "discourse motifs" representation method. By constructing a recursive hypergraph, they captured profound structural differences in texts, which significantly enhanced detection performance for cross-domain and long-text scenarios.

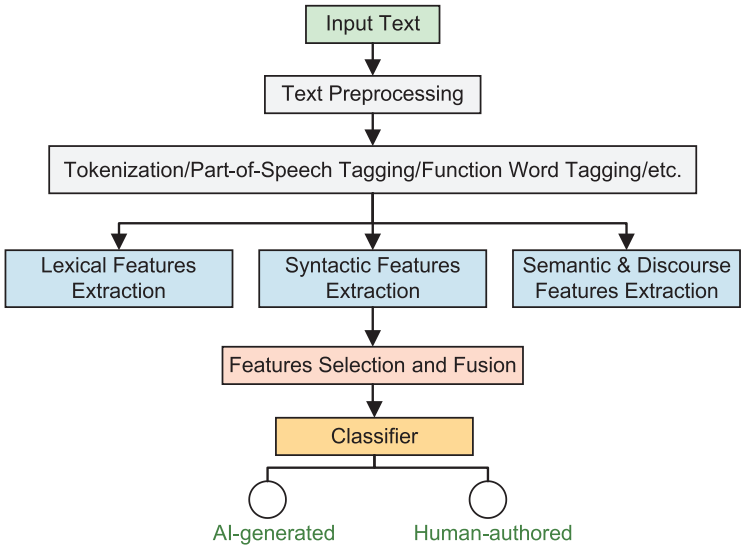

Figure 3: Flowchart of surface linguistic feature-based AI-generated text detection

While earlier surface linguistic feature-based methods performed well in specific scenarios, the increasing sophistication of LLMs has exposed the limitations of approaches that rely solely on shallow linguistic

features, such as their lack of robustness and poor cross-domain generalization. To address these challenges, research has begun to combine multiple features with deep learning models. For instance, Alhijawi et al. [28] proposed a multi-modal model named AI-Catcher for detecting ChatGPT-generated scientific text. This method fuses two distinct types of features: a multilayer perceptron (MLP) component learns representations of linguistic and statistical features, while a convolutional neural network (CNN) component extracts high-level sequential patterns from the textual content itself. By concatenating the two feature representations, AI-Catcher leverages information from different modalities, thereby significantly improving its detection accuracy in specific domains, such as scientific content. Their release of the AIGTxt dataset also provides a valuable benchmark for future evaluation.

In summary, surface linguistic feature-based detection methods have played a foundational role in AIGTD owing to their intuitive design, strong interpretability, and modest computational requirements. By leveraging lexical, syntactic, semantic, and discourse-level attributes, they can uncover systematic differences between human and AI-generated text, effects that were especially salient for earlier generators. Their transparency and capacity for visualization also make them attractive for sensitive application domains such as content moderation, educational integrity verification, and forensic linguistics.

Nevertheless, the rapid advance of LLMs—such as GPT-4, GPT-5, and DeepSeek—has increasingly exposed the limits of shallow linguistic features. Handcrafted indicators and statistical heuristics are fragile, highly domain-dependent, and susceptible to adversarial paraphrasing or stylistic manipulation, leading to degraded detection accuracy. Moreover, models trained on specific corpora or languages struggle to generalize to heterogeneous real-world contexts in which stylistic diversity and semantic nuance are critical. The redundancy and instability among manually engineered features further erode robustness and interpretability. These challenges underscore the need to move beyond shallow cues by integrating surface-linguistic feature approaches with deep neural representations and by developing semantically grounded, context-aware detection frameworks.

Table 2 summarizes the representative surface linguistic feature-based detection methods discussed in this section.

Table 2: Representative surface linguistic feature-based detection

Name	Ref.	Year	Key points
Linguistic feature ensemble	[20]	2021	Combines several shallow linguistic features with traditional classifiers (e.g., SVM, RF); evaluates how each feature and classifier uniquely contribute to the detection of LLM-generated text, tested on GPT-2, GPT-3, and Grover.
Syntactic-discourse profiling	[21]	2023	Compares POS tag distributions and discourse markers between AI-generated and human-authored texts, and trains a logistic regression classifier on stylistic features.
Linguistic-statistical classifier ensemble	[22]	2023	Constructs multiple classifiers (logistic regression, random forest, LightGBM, and BERT) to distinguish AI- and human-authored Chinese academic abstracts.

(Continued)

Table 2 (continued)

Name	Ref.	Year	Key points
Unsupervised and distributional detection	[23]	2021	Detects LLM-generated texts by leveraging repetitive high-order *n*-grams (super-maximal repeats) and ensemble classification without labeled data.
Neural authorship attribution framework	[24]	2020	Utilizes statistical and linguistic psychological features to analyze the differences between human-authored and AI-generated texts.
Coco	[25]	2022	Models surface-level coherence through sentence order perturbations and paragraph structure graphs, using contrastive learning for representation alignment.
xFakeSci	[26]	2023	Compares the differences in vocabulary importance distribution between AI-generated text and authentic scientific literature by calculating the TF-IDF weights of word pairs.
Discourse motif modeling	[27]	2024	Constructs a set of discourse motifs to represent subtle textual structures such as contrast, elaboration, and attribution.
AI-catcher	[28]	2025	Analyzes a combination of linguistic and statistical features, along with high-level sequential patterns from the text, to detect ChatGPT-generated scientific content.

3.2 Language Models-Based Detection

Language model-based detection methods leverage the probabilistic and contextual representations of pre-trained large language models to identify AI-generated text. Unlike surface linguistic feature-based approaches that depend on handcrafted cues, these methods directly exploit the internal scoring mechanisms of LLMs, such as token-level probabilities, entropy, or ranking statistics, to capture generative regularities.

As illustrated in Fig. 4, two dominant categories have emerged. Zero-shot detection operates without labeled training data by scoring input text with pre-trained models and applying statistical thresholds to determine the likelihood of a match. It enables rapid deployment but often suffers from sensitivity to model versions and unstable performance across domains. Fine-tuning-based detection, by contrast, adapts pre-trained models using labeled datasets, allowing them to learn discriminative representations tailored for classification. These detectors achieve higher accuracy and adaptability to specific tasks, but at the expense of substantial data requirements and vulnerability to distribution shifts.

3.2.1 Zero-Shot Detection

Zero-shot detection methods can identify AI-generated text without requiring additional training or fine-tuning. Instead of learning from labeled data, these methods exploit the internal statistical properties of large language models (LLMs) to identify generative patterns. A common strategy analyzes the token-level log-probabilities that an LLM assigns to an input text. Because AI-generated text typically exhibits

distinct probability distributions, such as smoother or more concentrated log-probability curves, zero-shot methods often calculate metrics like the curvature of the negative log-likelihood (NLL) or the variance of token ranks. A threshold is then applied to these metrics to distinguish between human-authored and AI-generated content.

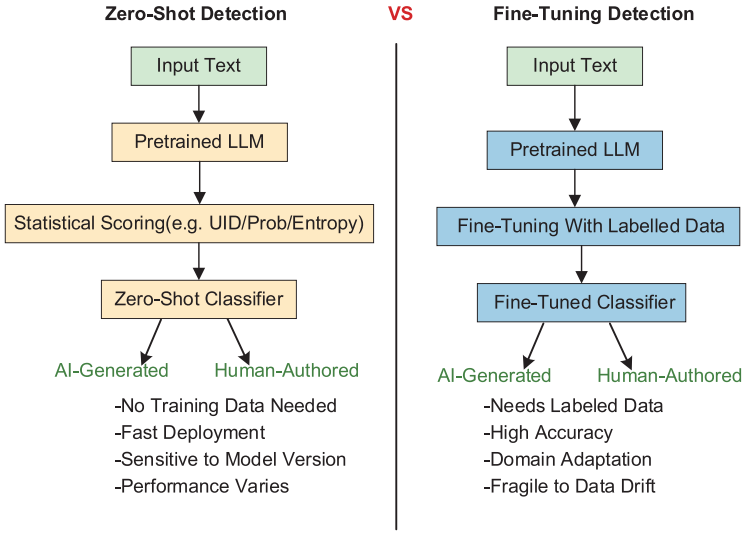

Figure 4: Comparison of zero-shot and fine-tuning-based AI-generated text detection methods

Early research showed that the probability distributions of AI-generated text follow statistical regularities within a language model. Gehrmann et al. [29] proposed the GLTR framework, which utilizes statistical indicators such as token prediction probability, rank, and entropy to reveal that generated text exhibits a notable tendency to use high-probability words. Subsequently, Mitchell et al. [19] introduced the DetectGPT method, which innovatively utilizes the probability curvature of a language model's output to distinguish between generated and natural text. They found that machine-generated text often lies in the region of negative curvature of the model's probability function. By perturbing the original text and analyzing the resulting changes in its log probability, they could achieve effective and highly robust detection. This method demonstrated strong cross-model adaptability on various open-source models (e.g., GPT-2, NeoX), achieving AUC scores that significantly outperformed most existing zero-shot methods.

However, DetectGPT's reliance on numerous perturbations incurs a high computational cost. To mitigate this, Miao et al. [30] proposed an improved method based on a Bayesian surrogate model. By using a Gaussian process to select a small number of representative perturbation samples, they achieved superior performance to DetectGPT with only 2–3 model queries, compared to the 200 queries used by the original. Bao et al. [31] introduced Fast-DetectGPT, which replaces the global perturbation strategy with conditional probability curvature. Fast-DetectGPT not only improved detection performance by approximately 75% in both white-box and black-box settings but also accelerated the detection process by a factor of 340, enabling detection within just 1–2 model calls. Furthermore, Liu et al. [32] noted that DetectGPT was prone to redundancy in perturbation selection and proposed the PECOLA framework. By combining selective perturbation with contrastive learning, the PECOLA framework effectively reduces noise and improves detection robustness in complex scenarios.

Beyond log probabilities, word-ranking signals have also proven useful. Su et al. [33] proposed DetectLLM, which incorporates two discriminative mechanisms based on log-likelihood and log-rank information. The perturbation-free variant efficiently calculates the ratio of log-likelihood to log-rank. In

contrast, the perturbation-sensitive variant leverages the source model's internal statistics by introducing a separate model (e.g., T5) to generate perturbations and analyzing the normalized change in log-rank. Experiments demonstrate that these methods strike a good balance between efficiency and performance, exhibiting strong generalization across various generative models, including GPT-2, OPT, and LLaMA.

Other approaches leverage multiple models. Hans et al. [34] introduced Binoculars, which uses the concept of cross-perplexity. By comparing a text's perplexity under one language model with its cross-perplexity under a second, Binoculars can efficiently detect AI-generated text without any training data. This process captures the statistical signature of generated text by analyzing its probability distribution across two different models (an observer and a performer). This methodology offers a novel technical approach for zero-shot detection, and its training-free nature makes it well-suited for detecting outputs from unknown models across diverse domains. While Binoculars excels at detecting distributional differences, its robustness in adversarial scenarios, such as paraphrasing, requires further investigation. Similarly, Yang et al. [35] proposed DNA-GPT, which truncates an original text, uses an LLM to generate a continuation, and then analyzes the divergence between the continuation and the original in terms of n-gram similarity and probability distribution. This training-free method performs stably on multilingual datasets (English and German) and is applicable in both white-box and black-box scenarios. Guo and Yu [36] proposed AuthentiGPT, which was the first method to leverage the denoising capability of language models for zero-shot detection. This approach injects noise into a text, uses a model like GPT-3.5-turbo to perform semantic denoising, and then compares the similarity between the denoised and original texts. The method does not require access to model parameters, making it suitable for black-box deployment, particularly for commercial API models like GPT-4.

While many zero-shot methods perform well on long texts, effectively detecting short LLM-generated texts remains challenging due to insufficient signal. To address this, Wei et al. [37] proposed Short-PHD, a zero-shot method specifically designed for short texts. Building on a prior work that utilizes persistent homology dimension (PHD) from topological data analysis, Short-PHD stabilizes the PHD estimation for short texts by inserting off-topic content before the input. This off-topic content insertion (OCI) technique helps alleviate the "local density peak" problem, making detection scores more stable and reliable. Through this novel approach, Short-PHD achieved superior performance in detecting short texts compared with existing zero-shot baselines and even some supervised methods.

In addition to proposing new detectors, another trend in zero-shot research focuses on improving the reliability and fairness of existing ones. Zhu et al. [38] introduced MCP (multiscale conformal prediction), a zero-shot framework designed to reliably constrain the false positive rate (FPR) while enhancing detection performance. Recognizing that a single, fixed threshold can lead to biased results across different text characteristics, MCP utilizes a small calibration set of human-authored texts to derive multiscale quantiles (thresholds) tailored to varying text lengths. By dynamically adjusting the threshold based on text length, MCP strikes a balance between FPR control and accuracy. This framework can be applied to a wide range of existing zero-shot detectors, demonstrating its strong generalization and potential to mitigate societal risks associated with misclassification.

Table 3 summarizes representative zero-shot-based detection methods.

Table 3: Representative zero-shot-based detection methods

Name	Ref.	Year	Key points
GLTR	[29]	2019	Visualizes token-level probability from a pretrained language model to reveal unnatural patterns in AI-generated text
DetectGPT	[19]	2023	Detects machine-generated text by using the logarithmic probability curvature of the perturbed samples.
Bayesian Surrogate model	[30]	2023	Improves the efficiency of DetectGPT by using a Bayesian surrogate model to approximate the curvature with fewer queries.
Fast-DetectGPT	[31]	2023	Applies conditional perturbation and approximated probability curvature to achieve rapid zero-shot detection with minimal model calls.
PECOLA	[32]	2024	Leverages perturbation-based contrastive views to build semantic representations for detecting AI-generated text, without requiring labeled examples or model fine-tuning.
DetectLLM	[33]	2023	Proposes a fast model-agnostic method using the log-rank test and likelihood ratio without needing perturbations or training.
Binoculars	[34]	2024	Compares the perplexity and cross-perplexity of text segments across models to detect AI generation without fine-tuning.
DNA-GPT	[35]	2023	Uses n-gram divergence between language model generations and original context to locate unnatural continuation boundaries.
AuthentGPT	[36]	2023	Leverages noise injection and semantic denoising to analyze reconstruction behavior and identify model-generated segments.
Short-PHD	[37]	2025	Stabilizes the persistent homology dimension (PHD) score by inserting off-topic content before the input text, making it effective for texts that are too short for other methods.
MCP	[38]	2025	Proposes a zero-shot framework that leverages multiscale conformal prediction to constrain the false positive rate (FPR), thereby enhancing the reliability and performance of existing detectors without additional training.

Overall, zero-shot methods provide efficiency and adaptability without requiring training data. Nonetheless, they face several challenges: (1) reliance on internal statistical properties of language models can hinder applicability to closed APIs; (2) adversarial perturbations—such as paraphrasing and style transfer—can substantially disrupt probability-based features, enabling many detectors to be evaded; and

(3) performance is sensitive to factors including model version, data domain, and text length, indicating that both robustness and interpretability warrant further improvement.

Future research may focus on hybrid scoring strategies, cross-model fusion, lightweight self-supervised training mechanisms, and adaptation techniques for multilingual and multi-style scenarios to enhance the applicability and scalability of zero-shot detection in real-world settings.

3.2.2 Fine-Tuning Detection

Unlike zero-shot methods, fine-tuning approaches adapt pre-trained language models (e.g., BERT, RoBERTa) into binary classifiers for text authenticity. These models are trained on large, labeled datasets to learn more targeted linguistic features. Typically, these methods take a text as input and output a probability or a class label ("human" or "AI"). Compared with traditional methods that rely on explicit textual features, fine-tuned language models can automatically learn more profound differences in semantics, syntax, and style, leading to stronger expressive power and higher detection accuracy. Typical tasks include classification at the document, paragraph, and sentence levels, offering high precision and flexible adaptation.

As AI-generated content becomes increasingly integrated into diverse contexts, fine-tuning methods have expanded from early document-level detection to multi-granularity tasks. In a technical evaluation, Popescu-Apreutesei et al. [39] demonstrated that a fine-tuned bidirectional long short-term memory (BiLSTM) model with a global vector for word representation embedding (GloVe) could achieve nearly 97% accuracy on the AI-generated abstracts (AI-GA) dataset. However, the authors emphasized that the inability to reach 100% accuracy highlights the ethical risks of relying solely on automated tools. They advocated for a hybrid approach that incorporates expert human judgement instead. To address mixed human-AI texts, Zeng et al. [40] proposed a hybrid-sentence RoBERTa classifier with a two-stage detection process involving both paragraph and sentence-level analysis. First, the text is segmented into paragraphs from different sources; then, a fine-grained judgment is made on the sentences within each paragraph. Models like RoBERTa are then fine-tuned on this basis to improve accuracy on such complex texts. Liu et al. [41] introduced ArguGPT, a multi-level detection framework for argumentative essays. By constructing a three-tiered classifier (document, paragraph, sentence), the model captures syntactic and lexical cues across granularities, achieving 99% accuracy at the document level. Oghaz et al. [42] conducted a systematic comparison of Transformer versus traditional classifiers, highlighting the superior performance of fine-tuned Transformer at both sentence-level and discourse-level classification. These studies collectively mark a shift from holistic discrimination to structural comprehension in fine-tuning detection approaches.

While refining detection tasks, researchers have also explored a diverse range of training strategies and model architectures for challenging inputs such as short texts, ambiguous boundaries, and domain-specific stylistic variations. For instance, Tian et al. [18] proposed MPU, a multi-scale positive unlabeled learning framework specifically designed for short text detection. By integrating length-sensitive loss functions with dynamic data augmentation, MPU addresses the partial labeling problem often encountered in real-world scenarios where annotated datasets are incomplete or imbalanced, improving robustness on concise textual inputs and adaptability to distribution shifts.

Mitrović et al. [43] adopted a complementary approach by integrating explainable AI (XAI) tools, such as SHAP, into their detection workflow. Through detailed analysis of keyword contributions within a DistilBERT model, they found that ChatGPT-generated text tends to adopt a neutral, formal tone with reduced lexical variety and minimal personalized expression. Such findings offered valuable insights into the stylistic discrimination mechanisms of language models and informed targeted fine-tuning strategies. In another line of work, Chen et al. [44] introduced the GPT-Sentinel framework, reframing detection as a

sequence-to-sequence prediction task. Using a T5 model to directly generate categorical labels ("human" or "AI"), their method leverages the generative capacity of Transformer models to capture nuanced contextual dependencies, resulting in notable gains in generalization and interpretability across a wide variety of benchmark datasets.

To address the poor robustness of existing detectors against out-of-distribution (OOD) data and adversarial attacks, as well as their lack of interpretable evidence, Chen et al. [45] proposed an IPAD (inverse prompt for AI detection) framework. The method adopts a multi-stage fine-tuning pipeline whose core is a Prompt Inverter that reconstructs an original prompt likely to have produced the given text; two distinguishers then evaluate the text's consistency with that prompt. IPAD demonstrates superior performance to existing baselines on in-distribution, OOD, and adversarial attack data. Furthermore, by providing evidence such as the predicted prompt and regenerated text, the framework significantly enhances the credibility and interpretability of its detection results.

Given the substantial variability in generator models, languages, and application domains observed in real-world settings, improving the transferability of fine-tuned detectors has become a major research focus. Rodriguez et al. [46] addressed this challenge through cross-domain detection, a two-step fine-tuning strategy. In the first stage, the model undergoes pre-training on proxy generator outputs to substitute for unseen generators. In the second stage, few-shot domain adaptation is performed to align the detector with the specific target domain. This incremental adaptation process was shown to alleviate domain shift issues significantly. Antoun et al. [47] extended this idea to multilingual contexts with XLM-R and CamemBERT models, showing stable detection accuracy across English, French, and Arabic. Furthermore, Wang et al. [48] developed a fine-tuned RoBERTa approach for fake news detection, training on social media platforms to distinguish human- from AI-generated content. Achieving 98% classification accuracy in noisy, real-world environments underscored the practical viability of such fine-tuning methods. Collectively, these studies highlight that fine-tuned detectors, when designed with adaptability and robustness in mind, can maintain strong performance even in scenarios involving high noise levels, varied linguistic registers, and rapidly evolving generative model behaviors.

Table 4 summarizes representative fine-tuning-based detection methods.

Table 4: Representative fine-tuning-based detection methods

Name	Ref.	Year	Key points
BiLSTM with GloVe	[39]	2025	Evaluates single-layer and dual-layer BiLSTM models on the AI-GA dataset, achieving approximately 97% accuracy and highlighting the ethical risks of relying solely on imperfect automated detectors.
Hybrid-sentence RoBERTa classifier	[40]	2024	Combines paragraph-level and sentence-level classification using a fine-tuned RoBERTa model to detect AI-generated content in human-AI collaborative writing.
ArguGPT	[41]	2023	Proposes a three-level detector to identify argumentative essays generated by GPT.
Transformer vs. Traditional classifiers	[42]	2025	Systematically comparing the detection performance of classic machine learning and Transformer models.

(Continued)

Table 4 (continued)

Name	Ref.	Year	Key points
MPU	[18]	2023	Proposes a multi-scale positive-unlabeled (MPU) detection framework by modeling short text detection as a partial positive-unlabeled problem.
SHAP	[43]	2023	Uses the SHAP method to interpret attention patterns and highlight the neutral tone of AI text.
GPT-sentinel	[44]	2023	Introduces the GPT-Sentinel framework with fine-tuned RoBERTa and T5 models to distinguish AI-generated from human-authored content.
IPAD	[45]	2025	Proposes a multi-stage fine-tuning framework that uses a Prompt Inverter to reconstruct original prompts and two distinguishers to evaluate text-prompt consistency.
Cross domain detection	[46]	2022	Proposes a cross-domain framework that fine-tunes RoBERTa with domain adaptation techniques to detect GPT-2-generated text across diverse topics.
Multilingual cross-LM detection	[47]	2023	Applies XLM-R and CamemBERT models to the datasets in English, French, and Arabic.
Fine-tuned RoBERTa for fake news	[48]	2023	Uses a fine-tuned RoBERTa model to detect AI-generated fake news on social media with 98% accuracy.

Despite advantages in accuracy, task adaptability, and semantic understanding, fine-tuning methods face several challenges. They depend heavily on large-scale, high-quality labeled data, which limits their applicability in low-resource scenarios. Fine-tuned models exhibit a strong dependency on the style of their training corpus, often suffering from performance decay when encountering different generators or new domains, a problem known as "in-domain overfitting". Finally, because these models are typically optimized for specific generators, their robustness may be insufficient against unseen models or novel generation strategies.

In contrast, zero-shot methods better accommodate diverse and rapidly iterating generators due to their training-free nature and flexibility, offering advantages in closed-API environments. However, these methods often rely on probability or structural outputs from the generator, which limits their effectiveness in black-box scenarios where this information is unavailable. In summary, fine-tuning methods offer superior accuracy and task-specific adaptation, whereas zero-shot methods excel in generalizability and deployment efficiency. Future research should focus on hybrid paradigms that integrate the efficiency of zero-shot scoring with the precision of fine-tuned detectors, employing strategies such as semi-supervised learning, transfer learning, and knowledge distillation to build more robust AIGTD systems.

4 Active Detection Methods

Active detection methods determine whether a text is AI-generated by embedding identifiable information during or immediately after generation. Compared with passive detection, active methods typically

require access to or control over the generation system, making them more suitable for platform-level deployment or proprietary model settings.

Broadly, active detection can be divided into two categories: (1) watermarking-based methods, which introduce embedded signals into the generated text to enable source attribution; and (2) retrieval-based methods, which compare the input text with a database of known model outputs based on semantic similarity. The following subsections provide a detailed introduction to these two active detection strategies.

4.1 Watermarking-Based Detection

The key idea behind watermarking-based AIGTD is to embed identifiable watermark signals into AI-generated text, allowing them to be later extracted and verified for reliable source attribution and traceability. A complete watermarking-based detection framework typically includes two interdependent phases: watermark embedding, where a specific watermark signal is embedded into the text during or after generation, and watermark detection, where the candidate text is subsequently analyzed to verify the presence of the embedded watermark signals.

The overall effectiveness of watermarking-based detection methods largely depends on the watermark embedding strategies, since they directly influence both the feasibility and robustness of subsequent detection. As shown in Fig. 5, watermarking embedding strategies can be categorized into two types according to the stage of the text generation pipeline where the watermark is embedded.

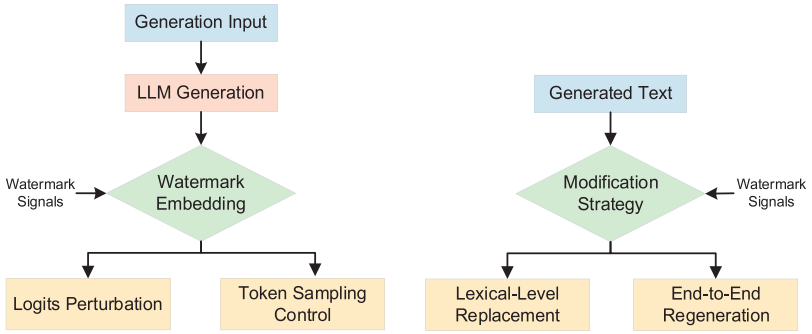

Figure 5: Embedding schemes of watermarking-based AIGTD

Generation-time watermarking (Fig. 5a) embeds watermark signals by incorporating watermarking operations into the model's generation process. This is typically implemented through two mechanisms: logits perturbation (e.g., modifying the output probability distribution via logit biasing or vocabulary partitioning) and token sampling control (e.g., entropy-gated selection or rule-based token promotion).

Post-generation watermarking (Fig. 5b) introduces watermark signals after different modification strategies have generated the text. The most common techniques include lexical-level replacement (e.g., context-aware synonym substitution) and end-to-end regeneration (e.g., semantic representation-based re-decoding). These operations are designed to preserve the original semantics while embedding identifiable watermark patterns.

Once the watermark embedding phase is complete, the detection process is applied to verify the presence of the watermark. As illustrated in Fig. 6, this process typically involves three stages: (1) watermark signal extraction, which identifies potential embedded cues from the candidate text; (2) a watermark score

assessment step, which quantifies the strength of the extracted signals into a numerical score; (3) a threshold-based decision, which determines whether the candidate text is watermarked AI-generated content or human-authored content.

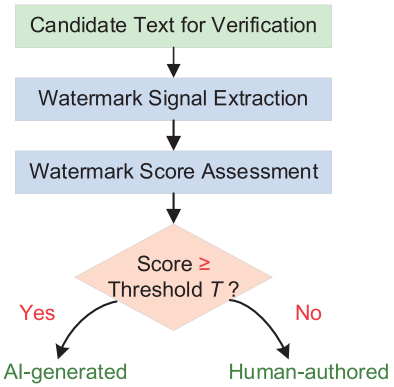

Figure 6: The verification phase of watermarking-based AIGTD

The design and implementation of watermark embedding approaches are crucial for enabling effective detection. Each method introduces a distinct statistical or semantic signature into the text, which in turn determines the watermark signal that must be identified during verification. The following sections elaborate on these embedding strategies, examining how their unique mechanisms ultimately support robust and efficient watermark detection.

In generation-time embedding, a typical approach intervenes in the token sampling distribution during logit generation. WLLM by Kirchenbauer et al. [49] exemplifies this: it uses a pseudorandom seed to partition the vocabulary. It adjusts the logits at each step, thereby increasing the probability of sampling "green" tokens and thus embedding a statistically identifiable bias. This strategy requires no access to model weights; detection relies on a hash function and hypothesis testing, making it efficient and straightforward. Building on this, Zhao et al. [50] proposed provable robust watermarking, which utilizes a fixed vocabulary partition rather than dynamic hashing to enhance robustness to text perturbations; however, this approach also increases the risk of the watermark being identified and attacked. For code generation, Lee et al. [51] introduced SWEET (selective watermarking with entropy thresholding), which inserts watermarks only at high-entropy positions. This enhances detection accuracy and imperceptibility while preserving the naturalness of the text. These methods typify perturbing the sampling process at the logit level.

Such strategies represent early watermarking paradigms focusing on token-level statistical manipulation. In addition, more advanced token-level methods have emerged. Chen et al. [52] introduced MCMARK (multi-channel-based watermarks), a family of unbiased, multi-channel-based watermarks that significantly enhance detectability and robustness while preserving text quality. MCMARK partitions the vocabulary into multiple segments and promotes token probabilities within a selected segment. This approach ensures that the original output distribution of the language model is maintained, addressing a key challenge of traditional biased watermarks. Furthermore, researchers are exploring synergistic frameworks that combine multiple strategies to enhance effectiveness. Wang et al. [53] proposed SymMark (symbiotic watermarking), a versatile symbiotic watermarking framework that aims to achieve synergy rather than a trade-off between robustness, text quality, and security. This method integrates both logits-based and sampling-based watermarking schemes. It adaptively embeds watermarks based on token entropy and semantic entropy, dynamically selecting the optimal approach for each token to ensure high detectability and robustness while minimizing impact on text quality.

However, more recent methods further leverage semantic-level constraints. In addition to modifying the logits, another class of generation-time watermarking methods focuses on embedding control during the token-sampling phase. Hou et al. [54] proposed the SemStamp method, which maps text to a vector space using sentence-level semantic encoding and uses a locality-sensitive hashing (LSH) structure to define watermark regions. Employing rejection sampling ensures that the generated content falls within a target semantic interval, thereby embedding a sentence-level semantic watermark. Christ et al. [55] approached the problem from a cryptographic perspective, constructing an undetectable watermarking system—a pseudorandom watermark structure controlled by a secret key. Its core lies in coupling a hash block with a key-based decoding mechanism. Although it lacks experimental support, it offers a new direction for the theoretical security of text watermarking. Beyond these semantic or cryptographic formulations within token-sampling control, Keleş et al. [56] introduced a sampling watermarker that samples multiple candidate tokens at each step, computes a secret hash-derived score for each, and selects the one with the highest score. This process embeds a detectable pattern without altering the model's probability distribution, achieving high detection rates and robustness against token-level paraphrasing. However, its performance is sensitive to sampling parameters, and its performance against more complex attacks remains to be validated.

In contrast to intervening directly in the model's sampling process, another category of methods processes the text after it has been generated, known as post-generation watermarking. Among these, synonym-level replacement methods hide information by substituting specific words in a way that preserves semantics. The word-level watermarking method based on contextual synonym replacement, proposed by Yang et al. [57], context-aware LS Watermarking, focuses on embedding watermarks at the lexical level while maintaining semantic integrity. Their early work uses a BERT model to generate semantically equivalent replacement words and a fine-tuned natural language inference model to screen for semantic consistency. By testing for synchronicity and substitutability, they identified suitable positions for embedding the watermark without altering the original meaning of the text. Subsequently, Yang et al. [58] introduced WTGB (watermarking for generated text in black-box models), a random binary encoding mechanism. This approach encodes candidate words into bits and replaces words representing '0' with words representing '1' while preserving semantic plausibility. Statistical hypothesis testing then distinguishes watermarked from standard text. This strategy achieves a more detectable watermark structure while maintaining semantic soundness, making it particularly suitable for black-box detection scenarios that do not require intervention in the generator's internal structure. Building on these methods, Hao et al. [59] introduced a post-hoc watermarking method for text generated by black-box language models. Their approach, RSFAW (robust semantics-faithful adaptive watermarking), embeds watermarks by identifying specific words that are semantically or syntactically fundamental to the text and therefore robust to minor modifications. The watermark is then injected using a paraphrase-based lexical substitution method that preserves the text's original semantic integrity. This strategy shows enhanced robustness against various attacks, including word substitution and paraphrasing, with higher detectability than previous methods.

In addition, REMARK-LLM, proposed by Zhang et al. [60], represents an end-to-end trainable watermarking method. This approach encodes the watermark signal into the semantic representation space during training and constructs a dedicated decoder for extraction. During generation, this decoder uses a beam search to regenerate watermarked natural language text from the latent vectors, creating a complete training-generation-detection loop. Table 5 lists representative watermarking-based detection methods.

Table 5: Representative watermarking-based detection methods

Name	Ref.	Year	Key points
WLLM	[49]	2023	Proposes a statistical method by randomly dividing the vocabulary and biasing the generation toward a "green list".
Provable robust watermarking	[50]	2023	Replaces dynamic vocabulary lists with fixed lists to enhance robustness, while potentially sacrificing stealthiness.
SWEET	[51]	2024	Inserts watermarks only at high-entropy positions in code generation tasks to improve imperceptibility and accuracy.
MCMARK	[52]	2025	Proposes an unbiased, multi-channel watermark that partitions the vocabulary into segments and promotes token probabilities within a selected segment based on a watermark key.
SymMark	[53]	2025	Proposes a symbiotic watermarking framework that integrates both logits-based and sampling-based schemes to achieve synergy between robustness, text quality, and security.
SemStamp	[54]	2024	Map sentences to a semantic space and use local similarity hashing technology to restrict the output to a predefined area.
Undetectable watermarking	[55]	2024	Defines theoretically undetectable watermarking technology using key-controlled hash blocks
Sampling watermarker	[56]	2025	Intervenes in the sampling process by selecting the token with the highest "secret number" from multiple candidates, embedding a pattern without altering the model's output distribution.
Context-aware LS watermarking	[57]	2022	Uses synonyms generated by BERT and the NLI model to ensure semantic consistency during the embedding process.
WTGB	[58]	2023	Proposes a decoupled framework to watermark text from black-box LMs via semantically-constrained lexical substitutions and a robust statistical detector, enabling watermarking without access to model internals.
RSFAW	[59]	2025	Injects watermarks using paraphrase-based lexical substitution on semantically or syntactically fundamental text features that are invariant to minor modifications, enhancing robustness and semantic faithfulness.
REMARK-LLM	[60]	2024	Encodes the watermark in the semantic representation and regenerates the text through a trained decoder.

Although watermarking technology provides a robust mechanism for content attribution, its robustness in complex adversarial environments remains a focal point for both academia and industry, particularly when facing paraphrasing attacks that preserve semantic meaning. To precisely quantify the vulnerability of AI-generated text detectors under paraphrasing attacks, Krishna et al. [17] developed a powerful paraphrasing model named DIPPER. This model provides a set of critical empirical data quantifying the performance degradation of standard watermarking methods proposed by Kirchenbauer et al. [49] under DIPPER attacks of varying strengths. Table 6 summarizes their core experimental findings, visually demonstrating how the accuracy of watermark detectors systematically declines as the strength of paraphrasing attacks increases.

Table 6: Impact of DIPPER paraphrasing attacks on watermark detection performance

Model	Attack type	Detection rate (%)
GPT-2-1.5B	No attack (baseline)	100.0
	+DIPPER (20L)	97.1
	+DIPPER (40L)	85.8
	+DIPPER (60L)	68.9
	+DIPPER (60L, 60O)	57.2
OPT-13B	No attack (baseline)	99.9
	+DIPPER (20L)	96.2
	+DIPPER (40L)	84.8
	+DIPPER (60L)	63.7
	+DIPPER (60L, 60O)	52.8

Note: Data is sourced from the study by Krishna et al. (2023) [17], showing the change in detection accuracy of the watermark detector under DIPPER paraphrasing attacks of varying intensities (L for lexical diversity, O for order diversity) at a fixed 1% False Positive Rate (FPR).

The data in Table 6 reveal a key issue: even minor, meaning-preserving paraphrases can significantly weaken the effectiveness of watermark detectors based on statistical signals.

Without any attack, the detection accuracy of the watermark is nearly perfect. However, as the intensity of the DIPPER attack increases, detection performance exhibits a marked and quantifiable decline. For example, with GPT-2-generated text, merely increasing lexical diversity (60L) reduces detection accuracy to 68.9%; when combined with content reordering (60L, 60O), accuracy drops further to 57.2%. A similar trend is observed for the larger OPT-13B model, where detection accuracy drops to 52.8% under the strongest attack.

Crucially, these attacks largely preserve the original text's semantics. The study by Krishna et al. reports that even under the strongest attack configuration, the semantic similarity score (Sim) between the paraphrased and original text remains above 94%. This indicates that an attacker can effectively erase or obscure the watermark signal without significantly altering the content's meaning, thereby successfully evading detection.

The success of the DIPPER attack demonstrates that leveraging external, high-capacity paraphrasing models is a highly effective and practical strategy for evading watermark detection, posing a severe challenge to the reliability of watermarking technology. Future research could further promote the integration of semantic-level embedding, neural watermarking mechanisms, and cross-modal detection, as well as exploring unified watermarking protocol standards to enhance the scalability and practicality of active detection systems.

4.2 Retrieval-Based Detection

As text-generation models expand in scale and capability, detecting AI-generated text has become increasingly challenging. Unlike passive detectors that infer authorship from intrinsic statistical properties of a single document (e.g., perplexity, token-level likelihood, stylometric cues), or active methods such as watermarking that verify an embedded watermark signal, retrieval-based detection reformulates the task as a provenance query: has the candidate text or a close semantic paraphrase appeared in the API's logged outputs? To address this, Krishna et al. [17] proposed a highly robust retrieval-based detection method, offering a new technical pathway for AIGTD. Rather than relying on local token statistics or watermark traces in the candidate alone, it performs a similarity search over an indexed log of prior generations. This design is more robust to paraphrasing as semantic content is preserved, is agnostic to model internals, and is auditable via matched evidence.

Fig. 7 illustrates a two-phase framework. Phase 1 (database construction): for each served prompt, the API's generated output is logged and indexed into a growing database that serves as the reference corpus for subsequent queries. Phase 2 (retrieval and verification): given a candidate text of unknown provenance, the system submits it to a retrieval module that performs a similarity search over the index. The maximum similarity among the matches is compared with a calibrated threshold, T. If the score $\geq T$, the detector reports "generated by this API"; otherwise, it returns "unknown."

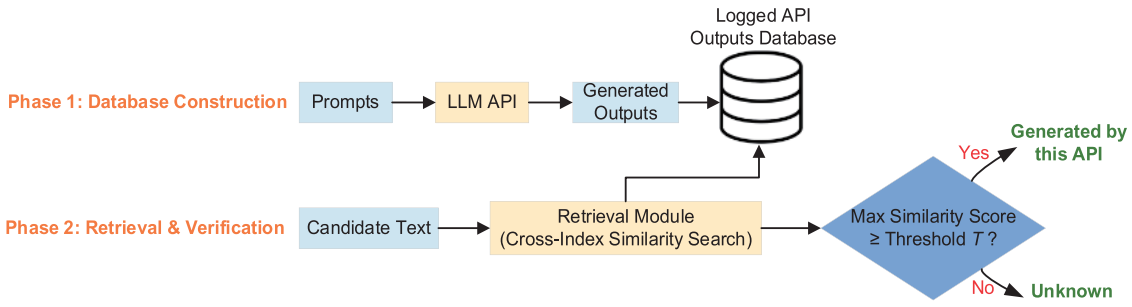

Figure 7: Retrieval-based detection framework

However, this method also has significant limitations. It relies on a complete archive of generated content; if data is not recorded at generation time, subsequent tracing is impossible. Furthermore, research by Sadasivan et al. [61] found that after five rounds of recursive paraphrasing, detection accuracy dropped below 60%, indicating that its robustness against extensive rewriting needs improvement. Additionally, retrieval-based methods face challenges related to data privacy and compliance in practical deployment. Regulations such as the European Union's General Data Protection Regulation (GDPR) may restrict how long model providers can store generated content, thereby affecting the feasibility of large-scale application of this method. Under the GDPR, storage must be time-bound and purpose-limited. Long retention improves recall but increases compliance risk; short retention reduces exposure but lowers coverage. Raw-text logs maximize auditability, yet create a high-value breach target and complicate erasure requests. De-identified embeddings reduce exposure, but still require deletion propagation and careful access control. In practice, systems should make this privacy–utility trade-off explicit and report retrieval recall as a function of the retention window.

Overall, retrieval-based detection, with its non-invasive nature, high interpretability, and resilience to light paraphrasing, serves as a valuable component of the active detection framework. Future research could explore the integration of adversarial training to enhance the robustness of semantic matching, develop

multi-round retrieval and reverse semantic restoration mechanisms, and create compliant detection systems that balance privacy with traceability under data protection regulations.

5 Leveraging Adversarial Learning to Enhance the Robustness of AIGTD

As the text generation capabilities of large language models continue to advance, a variety of evasion methods have emerged. Adversarial operations, such as minor text paraphrasing, semantic perturbation, and instruction rewriting, can compromise detection performance, raising false-positive rates and lowering detection rates. Traditional detectors often exhibit significant vulnerability to such adversarial perturbations. Therefore, enhancing the robustness and adaptability of detection systems in complex environments through the introduction of adversarial learning has become a critical research direction in AIGTD.

Early research demonstrated that even high-performing AIGTD detectors are susceptible to failure under minor perturbations. Crothers et al. [62] conducted a systematic evaluation at the feature level, finding that while neural network-based features excel in standard scenarios, statistical features exhibit greater robustness when subjected to character-level and word-level adversarial attacks (e.g., TextFooler, DeepWordBug). This revealed the sensitivity of deep features to superficial perturbations. To address more sophisticated paraphrasing techniques, Shi et al. [63] designed adversarial attack strategies based on synonym substitution and style transfer, showing that these light modifications can substantially degrade the performance of various detectors. Furthermore, Wang et al. [64] proposed a comprehensive stress-testing framework covering 12 attack methods across the pre-generation, in-generation, and post-generation stages, systematically evaluating the robustness of statistical detectors, fine-tuned language model detectors, and watermark-based detectors. The results showed that the average accuracy of all detectors dropped by approximately 35% under attack, and even the most robust watermarking methods had applicability limitations. Additionally, within their adversarial detection attack (ADAT) framework, Zhou et al. [65] generated adversarial samples that successfully evaded multiple detectors by using perturbation strategies based on word importance and perplexity, further confirming the vulnerability of current detection systems to fine-grained rewriting attacks.

Recent studies have further systematically improved the avoidance strategies based on synonymous sentences. Kadhim et al. [66] introduced an embedding-guided substitution method that prioritizes low-probability tokens while ensuring semantic consistency through WordNet synonyms. The approach employs an interpretable TM-AE embedding to control the replacements and demonstrated significant decreases in detection accuracy. Specifically, against Fast-DetectGPT, the AUROC on the XSum dataset dropped from 0.4431 to 0.2744 (a relative reduction of 38.1%), while on SQuAD it fell from 0.5068 to 0.3532 (a relative decrease of 30.3%). These findings highlight that semantically constrained rewrites remain a persistent threat, reinforcing the need for adversarial learning strategies to enhance robustness.

Collectively, these studies indicate that existing AIGTD systems lack adaptability to realistic adversarial scenarios and require the integration of systematic adversarial learning mechanisms to improve robustness. In response to these vulnerabilities, researchers have proposed various adversarial learning methods to enhance detectors, such as improving feature extraction, optimizing training procedures, or designing adversarial training schemes where attackers and defenders improve the model's resilience to attacks.

Shen et al. [67] proposed the TextDefense framework, which detects adversarial perturbations by leveraging changes in the entropy of the word importance distribution within a text. Because attacks typically increase entropy, TextDefense effectively captures this shift, achieving robust detection across multiple attack environments. For higher-level feature modeling, Huang et al. [68] introduced the siamese calibrated reconstruction network (SCRN). By incorporating a reconstruction network and a siamese calibration mechanism, SCRN separates the semantic and noise components of a text, significantly improving detector

performance against mild adversarial perturbations like PWWS and DeepWordBug. SCRN shows excellent adversarial robustness and generalization capabilities across multiple cross-domain datasets.

To more systematically enhance detector robustness against paraphrasing, Hu et al. [69] proposed the RADAR (robust AI-text detector via adversarial learning) framework. This framework co-trains a rewriter and a detector and uses the PPO algorithm to optimize their interaction. RADAR shows superior anti-paraphrasing ability and transferability on texts generated by eight different language models and across multiple domain datasets. Guo et al. [70] introduced the OUTFOX framework, which incorporates a context-interactive learning mechanism. Through continuous competition, it generates more deceptive adversarial texts, thereby promoting detector adaptation to novel attacks. OUTFOX achieved a top F1 score of 96.9 on a real-world student essay detection task, validating the potential of interactive adversarial learning in practical applications. Meanwhile, Zhou et al. [71] systematically developed 12 text perturbation methods of varying granularities. They proposed a comprehensive strategy combining multi-granularity adversarial sample training, dynamic adversarial training, contrastive learning, and domain adaptation. Although this approach achieved significant results against common perturbations, performance bottlenecks remain for unseen, extreme attack types, providing direction for future research.

Despite significant progress, multiple challenges persist. Many adversarial training methods are optimized for specific attack scenarios and lack universal defenses against complex, composite attacks. Performance still degrades on unseen domains or under cross-lingual transfer. Furthermore, adversarial training incurs high computational and data generation costs, limiting scalability in large-scale applications.

6 Conclusion

Parallel advances in detection technologies have accompanied the rapid evolution of AI-generated content. As this survey has demonstrated, the field of AI-generated text detection has progressed from early surface linguistic feature analysis to a mature methodological landscape, encompassing zero-shot detectors, fine-tuned models, watermarking techniques, and retrieval-based systems. Our taxonomy organizes these approaches into two fundamental paradigms: passive and active detection, which are defined by a critical trade-off between post-hoc analytical flexibility and source-level control. Despite substantial progress, increasingly sophisticated LLMs continue to expose limitations in both paradigms.

For passive detection, the primary challenge is susceptibility to semantic-preserving perturbations, such as paraphrasing, and limited generalization across domains or unseen generators like GPT-4, Claude, and DeepSeek, whose outputs often resemble human-authored text. Future research should move beyond shallow surface cues to capture deeper structural and semantic artifacts, strengthen robustness through advanced adversarial training and adaptive learning, and explore hybrid frameworks that combine automated detection with human expertise to address residual blind spots.

For active detection, key obstacles concern security, scalability, and practical deployment. Watermarking-based techniques offer stronger reliability by embedding signals at the source; however, they require access to the generation process and remain vulnerable to adaptive signal-removal or spoofing attacks. Moreover, considerations of privacy, regulatory compliance, and the computational overhead of storing and retrieving generation logs pose significant barriers to widespread adoption. Progress in this paradigm depends on designing more secure and imperceptible watermarking schemes, developing privacy-preserving protocols, and establishing collaborative infrastructures that operate effectively and ethically in real-world contexts.

By structuring the landscape around these paradigms, this survey provides more than a catalog of existing methods; it offers a strategic roadmap that clarifies challenges, highlights research gaps, and points to promising future directions for the field.

To address the challenges identified in this survey, future research may proceed along several key directions as follows:

(1) Cross-domain and cross-task detection: Develop detectors capable of transferring across domains, text styles, and outputs from diverse generators to improve adaptability in applications.
(2) Human–AI collaborative detection mechanisms: Explore frameworks that combine rapid automated detection with nuanced expert human review, leveraging human intuition to compensate for blind spots of machine detectors.
(3) Enhanced robustness: Strengthen resilience against adversarial attacks through advanced adversarial training, adaptive learning, and techniques for detecting previously unseen perturbations.
(4) Privacy-aware and compliant system design: Incorporate privacy protection and regulatory compliance into detector design to ensure feasibility and trustworthiness in practical deployments.

Through innovation across these dimensions, AIGTD can evolve from preliminary defense toward an intelligent, robust, and trustworthy framework for safeguarding digital information.

Acknowledgement: None.

Funding Statement: This work was supported in part by the Science and Technology Innovation Program of Hunan Province under Grant 2025RC3166, the National Natural Science Foundation of China under Grant 62572176, and the National Key R&D Program of China under Grant 2024YFF0618800.

Author Contributions: The authors confirm contribution to the paper as follows: Conceptualization, Lingyun Xiang; investigation, Nian Li; writing—original draft preparation, Nian Li; writing—review and editing, Jiayong Hu; supervision, Lingyun Xiang, Yuling Liu. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study.

References

1. Brown T, Mann B, Ryder N, Subbiah M, Kaplan JD, Dhariwal P, et al. Language models are few-shot learners. Adv Neural Inf Process Syst. 2020;33:1877–901. doi:10.5555/3495724.3495883.
2. Devlin J, Chang MW, Lee K, Toutanova K. Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Stroudsburg, PA, USA: Association for Computational Linguistics; 2019. p. 4171–86. doi:10.18653/v1/N19-1423.
3. Raffel C, Shazeer N, Roberts A, Lee K, Narang S, Matena M, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. J Mach Learn Res. 2020;21(140):5485–551. doi:10.5555/3455716.3455856.
4. Guo D, Yang D, Zhang H, Song J, Wang P, Zhu Q, et al. Deepseek-R1 incentivizes reasoning in LLMs through reinforcement learning. Nature. 2025;645(8081):633–8. doi:10.1038/s41586-025-09422-z.
5. Deshpande A, Murahari V, Rajpurohit T, Kalyan A, Narasimhan K. Toxicity in ChatGPT: analyzing persona-assigned language models. In: Findings of the Association for Computational Linguistics: EMNLP 2023. Stroudsburg, PA, USA: Association for Computational Linguistics; 2023. p. 1236–70. doi:10.18653/v1/2023.findings-emnlp.88.

6. Wen J, Ke P, Sun H, Zhang Z, Li C, Bai J, et al. Unveiling the implicit toxicity in large language models. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP 2023). Stroudsburg, PA, USA: Association for Computational Linguistics; 2023. p. 1322–38. doi:10.18653/v1/2023.emnlp-main.84.
7. Liang PP, Wu C, Morency LP, Salakhutdinov R. Towards understanding and mitigating social biases in language models. arXiv:2106.13219. 2021.
8. Palanivinayagam A, El-Bayeh CZ, Damaševičius R. Twenty years of machine-learning-based text classification: a systematic review. Algorithms. 2023;16(5):236. doi:10.3390/a16050236.
9. Li J, Xia Y, Cheng X, Zhao D, Yan R. Learning disentangled representation via domain adaptation for dialogue summarization. In: WWW '23: Proceedings of the ACM Web Conference 2023. New York, NY, USA: ACM; 2023. p. 1693–702. doi:10.1145/3543507.3583389.
10. Wiher G, Meister C, Cotterell R. On decoding strategies for neural text generators. Trans Assoc Comput Linguist. 2022;10:997–1012. doi:10.1162/tacl_a_00502.
11. Wei J, Tay Y, Bommasani R, Raffel C, Zoph B, Borgeaud S, et al. Emergent abilities of large language models. arXiv:2206.07682. 2022.
12. Lai V, Nguyen C, Ngo N, Nguyễn T, Dernoncourt F, Rossi R, et al. Okapi: instruction-tuned large language models in multiple languages with reinforcement learning from human feedback. In: Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. Stroudsburg, PA, USA: Association for Computational Linguistics; 2023. p. 318–27. doi:10.18653/v1/2023.emnlp-demo.28.
13. Farquhar S, Kossen J, Kuhn L, Gal Y. Detecting hallucinations in large language models using semantic entropy. Nature. 2024;630(8017):625–30. doi:10.1038/s41586-024-07421-0.
14. Li N, Li Y, Liu Y, Shi L, Wang K, Wang H. Drowzee: metamorphic testing for fact-conflicting hallucination detection in large language models. ACM Program Lang. 2024;8(OOPSLA2):1843–72. doi:10.1145/3689776.
15. Nabata KJ, Alshehri Y, Mashat A, Wiseman SM. Evaluating human ability to distinguish between ChatGPT-generated and original scientific abstracts. Updates Surg. 2025;77(3):615–21. doi:10.1007/s13304-025-02106-3.
16. Achiam J, Adler S, Agarwal S, Ahmad L, Akkaya I, Aleman FL, et al. GPT-4 technical report. arXiv:2303.08774. 2023.
17. Krishna K, Song Y, Karpinska M, Wieting J, Iyyer M. Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. New York, NY, USA: ACM; 2023. p. 27469–500. doi:10.5555/3666122.3667317.
18. Tian Y, Chen H, Wang X, Bai Z, Zhang Q, Li R, et al. Multiscale positive-unlabeled detection of AI-generated texts. arXiv:2305.18149. 2023.
19. Mitchell E, Lee Y, Khazatsky A, Manning CD, Finn C. DetectGPT: zero-shot machine-generated text detection using probability curvature. In: Proceedings of the 40th International Conference on Machine Learning. New York, NY, USA: ACM; 2023. p. 24950–62. doi:10.5555/3618408.3619446.
20. Fröhling L, Zubiaga A. Feature-based detection of automated language models: tackling GPT-2, GPT-3 and Grover. PeerJ Comput Sci. 2021;7(1):e443. doi:10.7717/peerj-cs.443.
21. Guo B, Zhang X, Wang Z, Jiang M, Nie J, Ding Y, et al. How close is ChatGPT to human experts? Comparison corpus, evaluation, and detection. arXiv:2301.07597. 2023.
22. Wang Y, Guo X, Liu Z, Wang J. Detection and comparative study of differences between AI-generated and scholar-written Chinese abstracts. J Intell. 2023;42(9):127–34. (In Chinese). doi:10.3969/j.issn.1002-1965.2023.09.018.
23. Gallé M, Rozen J, Kruszewski G, Elsahar H. Unsupervised and distributional detection of machine-generated text. arXiv:2111.02878. 2021. doi:10.48550/arXiv.2111.02878.
24. Uchendu A, Le T, Shu K, Lee D. Authorship attribution for neural text generation. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). Stroudsburg, PA, USA: Association for Computational Linguistics; 2020. p. 8384–95. doi:10.18653/v1/2020.emnlp-main.673.
25. Liu X, Zhang Z, Wang Y, Pu H, Lan Y, Shen C. Coco: coherence-enhanced machine-generated text detection under data limitation with contrastive learning. In: Proceedings of the 2023 Conference on Empirical Methods in Natural

Language Processing. Stroudsburg, PA, USA: Association for Computational Linguistics; 2023. p. 16167–88. doi:10.18653/v1/2023.emnlp-main.1005.
26. Hamed AA, Wu X. Detection of ChatGPT fake science with the xFakeSci learning algorithm. Sci Rep. 2024;14(1):16231. doi:10.1038/s41598-024-66784-6.
27. Kim ZM, Lee KH, Zhu P, Raheja V, Kang D. Threads of subtlety: detecting machine-generated texts through discourse motifs. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. p. 5449–74. doi:10.18653/v1/2024.acl-long.298.
28. Alhijawi B, Jarrar R, AbuAlRub A, Bader A. Deep learning detection method for large language models-generated scientific content. Neural Comput Appl. 2025;37(1):91–104. doi:10.1007/s00521-024-10538-y.
29. Gehrmann S, Strobelt H, Rush AM. GLTR: statistical detection and visualization of generated text. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations. Stroudsburg, PA, USA: Association for Computational Linguistics; 2019. p. 111–6. doi:10.18653/v1/P19-3019.
30. Miao Y, Gao H, Zhang H, Deng Z. Efficient detection of LLM-generated texts with a Bayesian surrogate model. In: Findings of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. p. 6118–30. doi:10.18653/v1/2024.findings-acl.366.
31. Bao G, Zhao Y, Teng Z, Yang L, Zhang Y. Fast-DetectGPT: efficient zero-shot detection of machine-generated text via conditional probability curvature. arXiv:2310.05130. 2023.
32. Liu S, Liu X, Wang Y, Cheng Z, Li C, Zhang Z, et al. Does DetectGPT fully utilize perturbation? Bridging selective perturbation to fine-tuned contrastive learning detector would be better. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. p. 1874–89. doi:10.18653/v1/2024.acl-long.103.
33. Su J, Zhuo TY, Wang D, Nakov P. DetectLLM: leveraging log-rank information for zero-shot detection of machine-generated text. In: Proceedings of the 2023 Findings of the Association for Computational Linguistics: EMNLP. Stroudsburg, PA, USA: Association for Computational Linguistics; 2023. p. 12395–412. doi:10.18653/v1/2023.findings-emnlp.827.
34. Hans A, Schwarzschild A, Cherepanova V, Kazemi H, Saha A, Goldblum M, et al. Spotting LLMs with binoculars: zero-shot detection of machine-generated text. In: Proceedings of the 41st International Conference on Machine Learning. New York, NY, USA: ACM; 2024. p. 17519–37. doi:10.5555/3692070.3692768.
35. Yang X, Cheng W, Wu Y, Petzold L, Wang WY, Chen H. DNA-GPT: divergent *n*-gram analysis for training-free detection of GPT-generated text. arXiv:2305.17359. 2023.
36. Guo Z, Yu S. AuthentiGPT: detecting machine-generated text via black-box language models denoising. arXiv:2311.07700. 2023.
37. Wei D, Mao M, Fang X, Chau M. Short-PHD: detecting short LLM-generated text with topological data analysis after off-topic content insertion. arXiv:2504.02873. 2025.
38. Zhu X, Ren Y, Cao Y, Lin X, Fang F, Li Y. Reliably bounding false positives: a zero-shot machine-generated text detection framework via multiscaled conformal prediction. In: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. p. 12298–319. doi:10.18653/v1/2025.acl-long.601.
39. Popescu-Apreutesei LE, Iosupescu MS, Necula SC, Păvăloaia VD. Upholding academic integrity amidst advanced language models: evaluating BiLSTM networks with GloVe embeddings for detecting AI-generated scientific abstracts. Comput Mater Contin. 2025;84(2):2605–44. doi:10.32604/cmc.2025.064747.
40. Zeng Z, Liu S, Sha L, Li Z, Yang K, Liu S, et al. Detecting AI-generated sentences in human-AI collaborative hybrid texts: challenges, strategies, and insights. In: Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence. New York, NY, USA: ACM; 2024. p. 7545–53.
41. Liu Y, Zhang Z, Zhang W, Yue S, Zhao X, Cheng X, et al. ArguGPT: evaluating, understanding, and identifying argumentative essays generated by GPT models. arXiv:2304.07666. 2023. doi:10.48550/arXiv.2304.07666.
42. Oghaz MM, Saheer LB, Dhame K, Singaram G. Detection and classification of ChatGPT-generated content using deep transformer models. Front Artif Intell. 2025;8:1458707. doi:10.3389/frai.2025.1458707.

43. Mitrović S, Andreoletti D, Ayoub O. ChatGPT or human? Detect and explain. Explaining decisions of a machine learning model for detecting short ChatGPT-generated text. arXiv:2301.13852. 2023.
44. Chen Y, Kang H, Zhai V, Li L, Singh R, Raj B. GPT-sentinel: distinguishing human and ChatGPT-generated content. arXiv:2305.07969. 2023.
45. Chen Z, Feng Y, He C, Deng Y, Pu H, Li B. IPAD: inverse prompt for AI detection—a robust and explainable LLM-generated text detector. arXiv:2502.15902. 2025. doi:10.48550/arXiv.2502.15902.
46. Rodriguez JD, Hay T, Gros D, Shamsi Z, Srinivasan R. Cross-domain detection of GPT-2-generated technical text. In: Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Stroudsburg, PA, USA: Association for Computational Linguistics; 2022. p. 1213–33. doi:10.18653/v1/2022.naacl-main.88.
47. Antoun W, Mouilleron V, Sagot B, Seddah D. Towards a robust detection of language model-generated text: is ChatGPT that easy to detect? arXiv:2306.05871. 2023.
48. Wang Z, Cheng J, Cui C, Yu C. Implementing BERT and fine-tuned RoBERTa to detect AI-generated news by ChatGPT. arXiv: 2306.07401.2023.
49. Kirchenbauer J, Geiping J, Wen Y, Katz J, Miers I, Goldstein T. A watermark for large language models. Proc Mach Learn Res. 2023;202:17061–84.
50. Zhao X, Ananth P, Li L, Wang YX. Provable robust watermarking for AI-generated text. arXiv:2306.17439. 2023.
51. Lee T, Hong S, Ahn J, Hong I, Lee H, Yun S, et al. Who wrote this code? Watermarking for code generation. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. p. 4890–911. doi:10.18653/v1/2024.acl-long.268.
52. Chen R, Wu Y, Guo J, Huang H. Improved unbiased watermark for large language models. In: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. p. 20587–601. doi:10.18653/v1/2025.acl-long.1005.
53. Wang Y, Ren Y, Cao Y, Fang B. From trade-OFF to synergy: a versatile symbiotic watermarking framework for large language models. In: Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. p. 10306–22. doi:10.18653/v1/2025.acl-long.509.
54. Hou A, Zhang J, He T, Wang Y, Chuang YS, Wang H, et al. SemStamp: a semantic watermark with paraphrastic robustness for text generation. In: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. p. 4067–82. doi:10.18653/v1/2024.naacl-long.226.
55. Christ M, Gunn S, Zamir O. Undetectable watermarks for language models. In: Proceedings of the Thirty-Seventh Annual Conference on Learning Theory; 2023 Jun 30–Jul 3; Edmonton, Canada. p. 1125–39.
56. Keleş KE, Gürbüz ÖK, Kutlu M. I know you did not write that! A sampling based watermarking method for identifying machine generated Text. In: Proceedings of the 1stWorkshop on GenAI Content Detection (GenAIDetect). Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. p. 140–9.
57. Yang X, Zhang J, Chen K, Zhang W, Ma Z, Wang F, et al. Tracing text provenance via context-aware lexical substitution. Proc AAAI Conf Artif Intell. 2022;36(10):11613–21. doi:10.1609/aaai.v36i10.21415.
58. Yang X, Chen K, Zhang W, Liu C, Qi Y, Zhang J, et al. Watermarking text generated by black-box language models. arXiv:2305.08883. 2023. doi:10.48550/arXiv.2305.08883.
59. Hao J, Qiang J, Zhu Y, Li Y, Yuan Y, Ouyang X. Post-hoc watermarking for robust detection in text generated by large language models. In: Proceedings of the 31st International Conference on Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2025. p. 5430–42.
60. Zhang R, Hussain SS, Neekhara P, Koushanfar F. REMARK-LLM: a robust and efficient watermarking framework for generative large language models. In: Proceedings of the 33rd USENIX Conference on Security Symposium. New York, NY, USA: ACM; 2024. p. 1813–30. doi:10.5555/3698900.3699002.
61. Sadasivan VS, Kumar A, Balasubramanian S, Wang W, Feizi S. Can AI-generated text be reliably detected? arXiv:2303.11156. 2023.

62. Crothers E, Japkowicz N, Viktor H, Branco P. Adversarial robustness of neural-statistical features in detection of generative transformers. In: Proceedings of the 2022 International Joint Conference on Neural Networks (IJCNN). New York, USA: IEEE; 2022. p. 1–8. doi:10.1109/IJCNN55064.2022.9892269.
63. Shi Z, Wang Y, Yin F, Chen X, Chang KW, Hsieh CJ. Red teaming language model detectors with language models. Trans Assoc Comput Linguist. 2024;12(8):174–89. doi:10.1162/tacl_a_00639.
64. Wang Y, Feng S, Hou A, Pu X, Shen C, Liu X, et al. Stumbling blocks: stress testing the robustness of machine-generated text detectors under attacks. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. p. 2894–925. doi:10.18653/v1/2024.acl-long.160.
65. Zhou Y, He B, Sun L. Humanizing machine-generated content: evading AI-text detection through adversarial attack. arXiv:2404.01907. 2024.
66. Kadhim AK, Jiao L, Shafik R, Granmo OC. Adversarial attacks on AI-generated text detection models: a token probability-based approach using embeddings. arXiv:2501.18998. 2025.
67. Shen L, Zhang X, Ji S, Pu Y, Ge C, Yang X, et al. Textdefense: adversarial text detection based on word importance entropy. arXiv:2302.05892; 2023.
68. Huang G, Zhang Y, Li Z, You Y, Wang M, Yang Z. Are AI-generated text detectors robust to adversarial perturbations?. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. p. 6005–24. doi:10.18653/v1/2024.acl-long.327.
69. Hu X, Chen PY, Ho TY. RADAR: robust AI-text detection via adversarial learning. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. New York, NY, USA: ACM; 2023. p. 15077–95. doi:10.5555/3666122.3666784.
70. Guo X, He Y, Zhang S, Zhang T, Feng W, Huang H, et al. Detective: detecting AI-generated text via multi-level contrastive learning. Adv Neural Inf Process Syst. 2024;t37:88320–47. doi:10.5555/3737916.3740718.
71. Zhou Y, He B, Sun L. Navigating the shadows: unveiling effective disturbances for modern AI content detectors. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics. Stroudsburg, PA, USA: Association for Computational Linguistics; 2024. p. 10847–61. doi:10.18653/v1/2024.acl-long.584.