DOI:10.1145/3624725

While many detection methods have been proposed, understanding the challenges is far more daunting.

BY RUIXIANG TANG, YU-NENG CHUANG, AND XIA HU

The Science of Detecting LLM-Generated Text

RECENT ADVANCEMENTS IN natural language generation (NLG) technology have significantly improved the diversity, control, and quality of large language models (LLM)-generated text. A notable example is OpenAI's ChatGPT, which demonstrates exceptional performance in tasks such as answering questions, composing email messages, essays, and codes. However, this newfound capability to produce human-like text at high efficiency also raises concerns about detecting and preventing misuse of LLMs in tasks such as phishing, disinformation, and academic dishonesty. For instance, many schools banned ChatGPT due to concerns over cheating in assignments,[11] and media outlets have raised the alarm over fake news generated by LLMs.[14] These concerns about the misuse of LLMs have hindered the NLG application in important domains such as media and education.

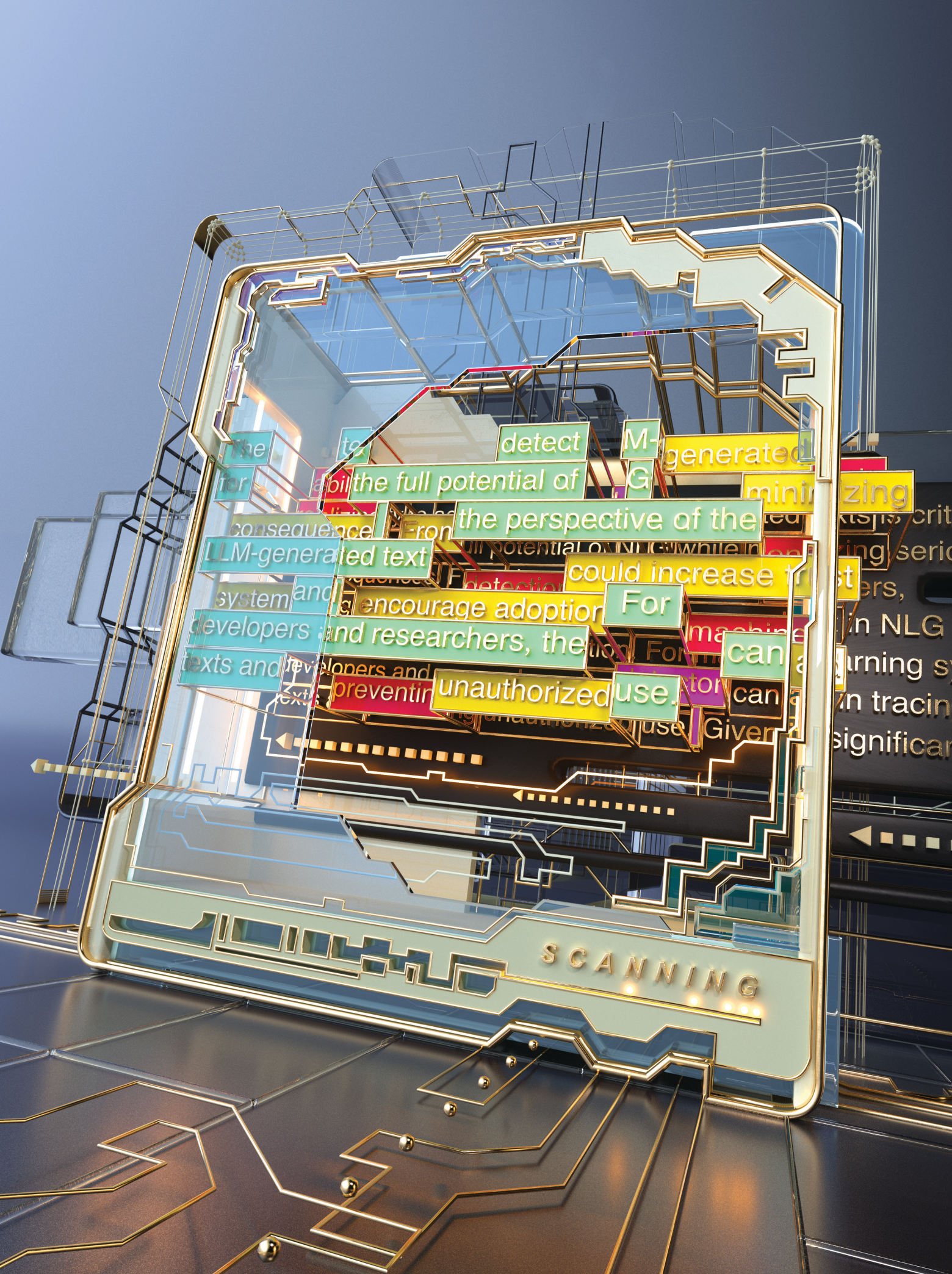
The ability to accurately detect LLM-generated text is critical for realizing the full potential of NLG while minimizing serious consequences. From the perspective of the end users, LLM-generated text detection could increase trust in NLG systems and encourage adoption. For machine learning system developers and researchers, the detector can aid in tracing generated text and preventing unauthorized use. Given its significance, there has been a growing interest in academia and industry to pursue research on LLM-generated text detection and to deepen our understanding of its underlying mechanisms.

While there is a rising discussion on whether LLM-generated text could be properly detected and how this can be done, we provide a comprehensive technical introduction of existing detection methods that can be grouped into two

» key insights

- **Existing LLM-generated text detection methods can be generally grouped into two categories: black-box detection and white-box detection. Black-box detection involves using API-level access to interact with and analyze LLM outputs. In contrast, white-box detection grants full access to the LLMs, enabling control over the model's generation behavior to enhance detectability.**
- **While black-box detection works at present due to detectable signals left by language models in generated text, it will gradually become less viable as language model capabilities advance and ultimately become infeasible.**
- **White-box detection methods are based upon the assumption that the LLM is controlled by the developers and offered as a service to end-users. However, the possibility of developers open-sourcing their LLMs poses a challenge to these detection approaches.**

ILLUSTRATION BY JACEY (DEBUT ART)


detect
the full potential of
the perspective of the
LLM-generated text
could increase
encourage adoption
For
and researchers, the
can
unauthorized
use.
SCANNING

general categories: black-box detection and white-box detection. Black-box detection methods are limited to API-level access to LLMs. They rely on collecting text samples from human and machine sources, respectively, to train a classification model that can be used to discriminate between LLM- and human-generated text. Black-box detectors work well because current LLM-generated text often show linguistic or statistical patterns. However, as LLMs evolve and improve, black-box methods are becoming less effective. An alternative is white-box detection: In this scenario, the detector has full access to the LLMs and can control the model's generation behavior for traceability purposes. In practice, black-box detectors are commonly constructed by external entities, whereas white-box detection is generally carried out by LLM developers.

This article is to discuss the timely topic from a data mining and natural language processing perspective. We first outline the black-box detection methods in terms of a data analytic life cycle, including data collection, feature selection, and classification model design. We then delve into more recent advancements in white-box detection methods, such as post-hoc watermarks and inference time watermarks. Finally, we present the limitations and concerns of current detection studies and suggest potential future research avenues. We aim to unleash the potential of powerful LLMs by providing fundamental concepts, algorithms, and case studies for detecting LLM-generated text.

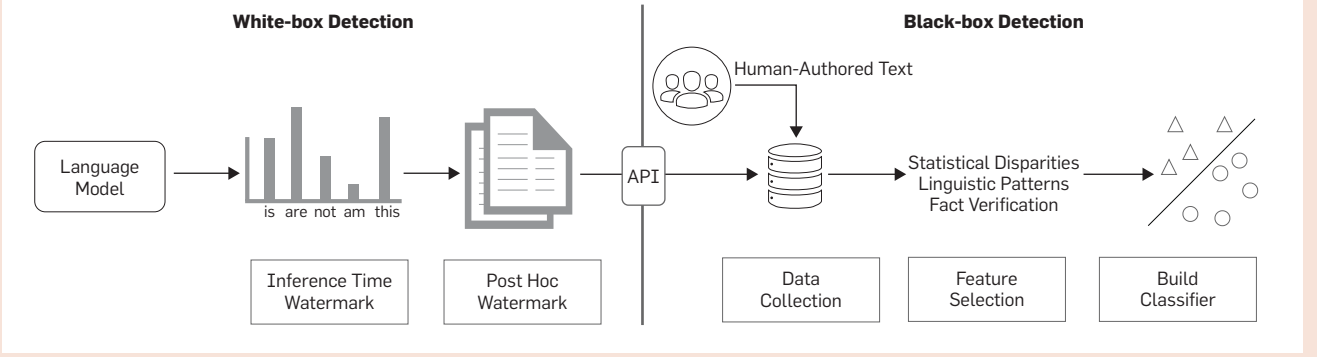
Prevalence and Impact

Recent advancements in LLMs, such as OpenAI's ChatGPT, have emphasized the potential impacts of this technology on individuals and society. Demonstrated through its performance on challenging tests, such as the MBA exams at Wharton Business School,[31] the capabilities of ChatGPT suggest its potential to provide professional assistance across various disciplines. In the healthcare domain, for example, the applications of ChatGPT extend far beyond simple enhancements in efficiency. ChatGPT not only optimizes documentation procedures, facilitating the generation of medical records, progress reports, and discharge summaries, it aids in the collection and analysis of patient data, facilitating medical professionals in making informed decisions regarding patient care. Recent research has also indicated the potential of LLMs in generating synthetic data for the healthcare field, thereby potentially addressing common privacy issues during data collection. The influence of LLM is also felt in the legal sector, where they are reshaping traditional practices such as contract generation and litigation procedures. The improved efficiency and effectiveness of these models are changing how we do things across a multitude of domains. As a result, when we talk about LLMs, we are not just measuring their technical competency, but also looking at their broader societal and professional implications.

The introduction of LLMs in education has elicited huge concerns. While convenient, their potential to provide quick answers threatens to undermine the development of critical thinking and problem-solving skills, which are essential for academic and life-long success. Further, there's the issue of academic honesty, as students might be tempted to use these tools inappropriately. In response, New York City public schools have prohibited the use of ChatGPT.[11] While the impact of LLMs on education is significant, it is imperative to extend this discourse to other domains. For instance, in journalism, the emergence of AI-generated "deepfake" news articles can threaten the credibility of news outlets and misinform the public. In the legal sector, the potential misuse of LLMs could have repercussions on the justice system, from contract generation to litigation processes. In cybersecurity, LLMs could be weaponized to create more convincing phishing emails or social engineering attacks.

To mitigate the potential misuse of LLMs, detection systems for LLM-generated text are emerging as a significant countermeasure. These systems offer the capability to differentiate AI-generated content from human-authored text, thereby playing a pivotal role in preserving the integrity of various domains. In the realm of academia, such tools can facilitate the identification of academic misconduct. Within the field of journalism, these systems may assist in separating legitimate news from AI-generated misinformation. Furthermore, in cybersecurity, they have the potential to strengthen spam filters to better identify and flag AI-aided threats. A recent incident at Texas A&M University underscores the urgent need for effective LLM detection tools.[10] An instructor suspected students of using ChatGPT to complete their final assignments. Lacking a detection tool, the instructor resorted to pasting the student's responses into ChatGPT, asking the ChatGPT if it had generated the text. This ad-hoc method sparked substantial debate online, il-

Figure 1. An overview of the LLM-generated text detection.



lustrating the pressing need for more sophisticated and reliable ways to detect LLM-generated content. While the current detection tools may not be flawless, they nonetheless symbolize a proactive effort to maintain ethical standards in the face of rapid AI advancements. The surge of interest in research focused on LLM-generated text detection testifies to the importance of these tools in mitigating the societal impact of LLMs. As such, we must conduct more extensive discussions on the detection of LLM-generated text. Particularly, we must explore its potential to safeguard the integrity of various domains against the risks posed by LLM misuse.

To mitigate the potential misuse of LLMs, detection systems for LLM-generated text are emerging as a significant countermeasure.

Black-Box Detection

In the domain of black-box detection, external entities are restricted to API-level access to the LLM, as depicted in Figure 1. To develop a proficient detector, black-box approaches necessitate gathering text samples originating from both human and machine-generated sources. Subsequently, a classifier is then designed to distinguish between the two categories by identifying and leveraging relevant features. We highlight the three essential components of black-box text detection: data acquisition, feature selection, and the execution of the classification model.

Data acquisition. The effectiveness of black-box detection models is heavily dependent on the quality and diversity of the acquired data. Recently, a growing body of research has concentrated on amassing responses generated by LLMs and comparing them with human-composed text spanning a wide range of domains. This section delves into the various strategies for obtaining data from both human and machine sources.

LLM-generated data. LLMs are designed to estimate the likelihood of subsequent tokens within a sequence, based on the preceding words. Recent advancements in natural language generation have led to the development of LLMs for various domains, including question-answering, news generation, and story creation. Prior to acquiring language model (LM)-generated text, it is essential to delineate target domains and generation models. Typically, a detection model is constructed to recognize text generated from a specific LM across multiple domains. To enhance detection generalizability, the minimax strategy suggests that detectors should minimize worst-case performance, which entails enhancing detection capabilities for the most challenging instances where the quality of LM-generated text closely resembles human-authored content.[27]

Generating high-quality text in a particular domain can be achieved by fine-tuning LMs on task-related data, which substantially improves the quality of the generated text. For example, Solaiman et al. fine-tuned the GPT-2 model on Amazon product reviews, producing reviews with a style consistent with those found on Amazon.[34] Moreover, LMs are known to produce artifacts such as repetitiveness, which can negatively impact the generalizability of the detection model. To mitigate these artifacts, researchers can provide domain-specific prompts or constraints before generating outputs. For instance, Clark et al.[6] randomly selected 50 articles from Newspaper3k to use as prompts for the GPT-3 model for news generation and applied filtering constraints on the models with the phrase "Once upon a time" for story creation. The token sampling strategy also significantly influences the generated text quality and style. While deterministic greedy algorithms like beam search[35] generate the most probable sequence, they may restrict creativity and language diversity. Conversely, stochastic algorithms such as nucleus sampling[19] maintain a degree of randomness while excluding inferior candidates, making them more suitable for a free-form generation. In conclusion, it is crucial for researchers to carefully consider the target domain, generation models, and sampling strategies when collecting LM-generated text to ensure the production of high-quality, diverse, and domain-appropriate content.

Human-authored data. Manual composition by humans serves as a natural method for obtaining authentic, human-authored data. For example, in a study conducted by Dugan et al.,[8] the authors sought to evaluate the quality of NLG systems and gauge human perceptions of the generated text. To achieve this, they employed 200 Amazon Mechanical Turk workers to complete 10 annotations on a website, accompanied by a natural language rationale for their choices. However, manually collect-

ing data through human effort can be both time-consuming and financially impractical for larger datasets. An alternative strategy involves extracting text directly from human-authored sources, such as websites and scholarly articles. For instance, we can readily amass thousands of descriptions of computer science concepts from Wikipedia, penned by knowledgeable human experts.[18] Moreover, numerous publicly accessible benchmark datasets, like ELI5,[13] which comprises 270K threads from the Reddit forum "Explain Like I'm Five," already offer human-authored text in an organized format. Utilizing these readily available sources can considerably decrease the time and expense involved in collecting human-authored text. Nevertheless, it is essential to address potential sampling biases and ensure topic diversity by including text from various groups of people, as well as non-native speakers.

Human evaluation findings. Prior research has offered valuable perspectives on differentiating LLM-generated text from human-authored text through human evaluations. Initial observations indicate that LLM-generated text are less emotional and objective compared to human-authored text, which often uses punctuation and grammar to convey subjective feelings.[18] For example, human authors frequently use exclamation marks, question marks, and ellipsis to express their emotions, while LLMs generate answers that are more formal and structured. However, it is crucial to acknowledge that LLM-generated text may not always be accurate or beneficial, as they can contain fabricated information.[18,33] At the sentence level, research has shown that human-authored text are more coherent than LLM-generated text, which tends to repeat terms within a paragraph.[9,28] These observations suggest that LLMs may leave some distinctive signals in their generated text, allowing for the selection of suitable features to distinguish between LLM and human-authored text.

Figure 2. Visualization results of GLTR,[17] the word ranking is obtained from the GPT-2 small model. Words that rank within the top 10 are highlighted in green, top 100 in yellow, top 1,000 in red, and the rest in purple. There is a notable difference between the two texts. The human-authored text are from Chalkbeat New York.[11]

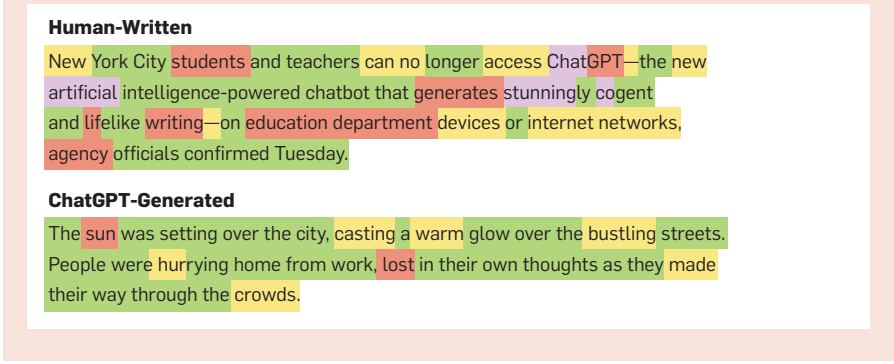
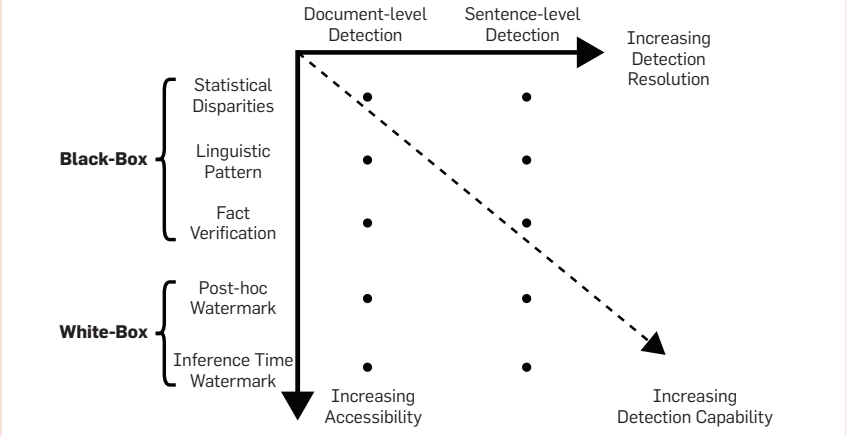


Figure 3. A taxonomy of LLM-generated text detection.



Detection feature selection. How can we discern between LM-generated text and human-authored text? This section will discuss possible detection features from multiple angles, including statistical disparities, linguistic patterns, and fact verification.

Statistical disparities. The detection of statistical disparities between LLM-generated and human-authored text can be accomplished by employing various statistical metrics. For example, the Zipfian coefficient measures the text's conformity to an exponential curve, which is described by Zipf's law.[29] A visualization tool, GLTR,[17] has been developed to detect generation artifacts across prevalent sampling methods, as demonstrated in Figure 2. The underlying assumption is that most systems sample from the head of the distribution, thus word ranking information of the language model can be used to distinguish LLM-generated text. Perplexity serves as another widely used metric for LLM-generated text detection. It measures the degree of uncertainty or surprise in predicting the next word in a sequence, based on the preceding words, by calculating the negative average log-likelihood of the text under the language model.[5]

Research indicates that language models tend to concentrate on common patterns in the text they were trained on, leading to low perplexity scores for LLM-generated text. In contrast, human authors possess the capacity to express themselves in a wide range of styles, making prediction more difficult for language models and resulting in higher perplexity values for human-authored text. However, it is important to recognize that these statistical disparities are constrained by the necessity for document-level text, which inevitably diminishes the detection resolution, as depicted in Figure 3.

Linguistic patterns. Various contextual properties can be employed to analyze linguistic patterns in human and LLM-generated text, such as vocabulary features, part-of-speech, dependency parsing, sentiment analysis, and stylistic features. The vocabulary features offer insight into the queried text's word usage patterns by analyzing charac-

teristics, such as average word length, vocabulary size, and word density. Previous studies on ChatGPT have shown that human-authored text tend to have a more diverse vocabulary but shorter length.[18] Part-of-speech analysis emphasizes the dominance of nouns in ChatGPT text, implying argumentativeness and objectivity, while the dependency parsing analysis shows that ChatGPT text use more determiners, conjunctions, and auxiliary relations.[18] Sentiment analysis, on the other hand, provides a measure of the emotional tone and mood expressed in the text.

Unlike humans, LLMs tend to be neutral by default and lack emotional expression. Research has shown that ChatGPT expresses significantly less negative emotion and hate speech compared to human-authored text. Stylistic features or stylometry, including repetitiveness, lack of purpose, and readability, are also known to harbor valuable signals for detecting LLM-generated text.[15] In addition to analyzing single text, numerous linguistic patterns can be found in multi-turn conversations.[3] These linguistic patterns reflect the training data and strategies of LLMs and serve as valuable features for detecting LLM-generated text. However, it is important to note that LLMs can substantially alter their linguistic patterns in response to prompts. For instance, incorporating a prompt like "Please respond with humor" can change the sentiment and style of the LLM's response, impacting the robustness of linguistic patterns.

Fact verification. LLMs often rely on likelihood maximization objectives during training, which can result in the generation of nonsensical or inconsistent text, a phenomenon known as hallucination. This emphasizes the significance of fact-verification as a crucial feature for detection.[40] For instance, OpenAI's ChatGPT has been reported to generate false scientific abstracts and post misleading news opinions. Studies also revealed that popular decoding methods, such as top-k and nucleus sampling, resulted in more diverse and less repetitive generations. However, they also produce text that are less verifiable.[25] These findings underscore the potential for using fact verification to detect LLM-generated text.

Prior research has advanced the development of tools and algorithms for conducting fact verification, which entails retrieving evidence for claims, evaluating consistency and relevance, and detecting inconsistencies in text. One strategy employs sentence-level evidence, such as extracting facts from Wikipedia, to directly verify facts of a sentence.[25] Another approach involves analyzing document-level evidence via graph structures, which capture the factual structure of the document as an entity graph. This graph is utilized to learn sentence representations with a graph neural network, followed by the composition of sentence representations into a document representation for fact verification.[40] Some studies also use knowledge graphs constructed from truth sources, such as Wikipedia, to conduct fact verification.[33] These methods evaluate consistency by querying subgraphs and identify non-factual information by iterating through entities and relations. Given that human-authored text may also contain misinformation, it is vital to supplement the detection results with other features to accurately distinguish text generated by LLMs.

Classification model. The detection task is typically approached as a binary classification problem, aiming to capture textual features that differentiate between human-authored and LLM-generated text. This section provides an overview of the primary categories of classification models.

Traditional classification algorithms utilize various features outlined previously to differentiate between human-authored and LLM-generated text. Some of the commonly used algorithms are support vector machines, naive Bayes, and decision trees. For instance, Fröhling et al. utilized linear regression, SVM, and random forests models built on statistical and linguistic features to successfully identify text generated by GPT-2, GPT-3, and Grover models.[15] Similarly, Solaiman et al. achieved solid performance in identifying text generated by GPT-2 through a combination of TF-IDF unigram and bigram features with a logistic regression model.[34] In addition, studies have also shown that using pre-trained language models to extract semantic textual features, followed by SVM for classification, can outperform the use of statistical features alone.[7] One advantage of these algorithms is their interpretability, allowing researchers to analyze the importance of input features and understand why the model classifies text as LLM-generated or not.

Deep learning approaches. In addition to employing explicitly extracted features for detection, recent studies have explored leveraging language models, such as RoBERTa,[24] as a foundation. This approach entails fine-tuning these language models on a mixture of human-authored and LLM-generated text, enabling the implicit capture of textual distinctions. Most studies adopt the supervised learning paradigm for training the language model, as demonstrated by Ippolito et al.,[20] who fine-tuned the BERT model on a curated dataset of generated-text pairs. Their study revealed that human raters have significantly lower accuracy than automatic discriminators in identifying LLM-generated text. In a low-resource scenario, Rodriguez et al.[30] showed that a few hundred labeled in-domain authentic and synthetic text suffice for robust performance, even without complete information about the LLM text-generation pipeline.

Despite the strong performance under the supervised learning paradigms, obtaining annotations for detection data can be challenging in real-world applications, leading the supervised paradigms inapplicable in some cases. Recent research[16] addresses this issue by detecting LLM-generated documents using repeated higher-order n-grams, which can be trained under unsupervised learning paradigms without requiring LLM-generated datasets as training data. Besides using the language model as the backbone, recent research finds that contextual structure can be viewed as a graph containing entities mentioned in the text and the semantically relevant relations, which utilizes a deep graph neural network to capture the structure feature of a document for LLM-generated news detection.[40] While deep learning approaches often yield superior detection outcomes, their black-box nature severely restricts interpretability. Consequently, researchers typically rely on interpretation tools to comprehend the rationale behind the model's decisions.

White-Box Detection

In white-box detection, the detector processes complete access to the target language model, facilitating the inte-

gration of concealed watermarks into its outputs to monitor suspicious or unauthorized activities. Here, we initially outline the three prerequisites for watermarks in NLG. Subsequently, we provide a synopsis of the two primary classifications of white-box watermarking strategies: post-hoc watermarking and inference-time watermarking.

Watermarking requirements. Building on prior research in traditional digital watermarking, we put forth three crucial requirements for NLG watermarking: *Effectiveness*: The watermark must be effectively embedded into the generated text and verifiable while preserving the quality of the generated text. *Secrecy:* To achieve stealthiness, the watermark should be designed without introducing conspicuous alterations that could be readily detected by automated classifiers. Ideally, it should be indistinguishable from non-watermarked text. *Robustness:* The watermark ought to be resilient and difficult to remove through common modifications such as synonym substitution. To eliminate the watermark, attackers must implement significant modifications that render the text unusable. These three requirements form the bedrock for NLG watermarking and guarantee the traceability of LLM-generated text.

Post-hoc watermarking. Given an LLM-generated text, post-hoc watermarks will embed a hidden message or identifier into the text. Verification of the watermark can be performed by recovering the hidden message from the suspicious text. There are two main categories of post-hoc watermarking methods: rule-based and neural-based approaches.

Rule-based approaches. Initially, nature language researchers adapted techniques from multimedia watermarking, which were non-linguistic in nature and relied heavily on character changes. For example, the line-shift watermark method involves moving a line of text upward or downward (or left or right) based on the binary signal (watermark) to be inserted.[4] However, these "printed text" watermarking approaches had limited applicability and were not robust against text reformatting. Later research shifted toward using the syntactic structure for watermarking. A study by Atallah et al.[2] embedded watermarks in parsed syntactic tree structures, preserving the meaning of the original text and rendering watermarks indecipherable to those without knowledge of the modified tree structure. In addition, syntactic tree structures are difficult to remove through editing and remain effective when the text is translated into other languages. Further improvements were made in a series of works, which proposed variants of the method that embedded watermarks based on synonym tables instead of just parse trees.[21]

While deep learning approaches often yield superior detection outcomes, their black-box nature severely restricts interpretability.

Along with syntactic structure, researchers have also leveraged the semantic. structure of text to embed watermarks. This includes exploiting features such as verbs, nouns, prepositions, spelling, acronyms, and grammar rules. For instance, a synonym substitution approach was proposed in which watermarks are embedded by replacing certain words with their synonyms without altering the context of the text.[36] Generally, rule-based methods utilize fixed rule-based substitutions, which may systematically change the text statistics, compromising the watermark's secrecy and allowing adversaries to detect and remove the watermark.

Neural-based approaches. In contrast to the rule-based methods that demand significant engineering efforts for design, neural-based approaches envision the information-hiding process as an end-to-end learning process. Typically, these approaches involve three components: a watermark encoder network, a watermark decoder network, and a discriminator network.[1] Given a target text and a secret message (for example, random binary bits), the watermark encoder network generates a modified text that incorporates the secret message. The watermark decoder network then endeavors to retrieve the secret message from the modified text. One challenge is the watermark encoder network may significantly alter the language statistics. To address this problem, the framework employs an adversarial training strategy and includes the discriminator network. The discriminator network takes the target text and watermarked text as input and aims to differentiate between them, while the watermark encoder network aims to make them indistinguishable. The training process continues until the three components achieve a satisfactory level of performance. For watermarking LLM-

generated text, developers can use the watermark encoder network to embed a preset secret message into LLMs' outputs, and the watermark decoder network to verify suspicious text. Although neural-based approaches eliminate the need for manual rule design, their inherent lack of interpretability raises concerns regarding their truthfulness and the absence of mathematical guarantees for the watermark's effectiveness, secrecy, and robustness.

Inference-time watermarking. Inference-time watermarking targets the LLM decoding process, as opposed to post-hoc watermarks, which are applied after text generation. The language model produces a probability distribution for the subsequent word in a sequence based on preceding words. A decoding strategy, which is an algorithm that chooses words from this distribution to create a sequence, offers an opportunity to embed the watermark by modifying the word selection process.

A representative example of this method can be found in research conducted by Kirchenbauer et al.[23] During the next token generation, a hash code is generated based on the previously generated token, which is then used to seed a random number generator. This seed randomly divides the whole vocabulary into a "green list" and a "red list" of equal size. The next token is subsequently generated from the green list. In this way, the watermark is embedded into every generated word, as depicted in Figure 4. To detect the watermark, a third party with knowledge of the hash function and random number generator can reproduce the red list for each token and count the number of violations of the red list rule, thus verifying the authenticity of the text. The probability that a natural source produces N tokens without violating the red list rule is only $1/2^N$, which is vanishingly small even for text fragments with a few dozen words. To remove the watermark, adversaries must modify at least half of the document's tokens. However, one concern with these inference-time watermarks is that the controlled sampling process may significantly impact the quality of the generated text. One solution is to relax the watermarking constraints, for example, increasing the green list vocabulary size, and aim for a balance between watermarking and text quality.

Benchmarking Datasets

This section introduces several noteworthy benchmarking datasets for detecting LLM-generated text. One prominent example is the work by Guo et al.,[18] where the authors constructed the dataset Human ChatGPT Comparison Corpus (HC3), specifically designed to distinguish text generated by ChatGPT in question-answering tasks in both English and Chinese languages. The dataset comprises 37,175 questions across diverse domains, such as open-domain, computer science, finance, medicine, law, and psychology. These questions and corresponding human answers were sourced from publicly available question-answering datasets and wiki text.

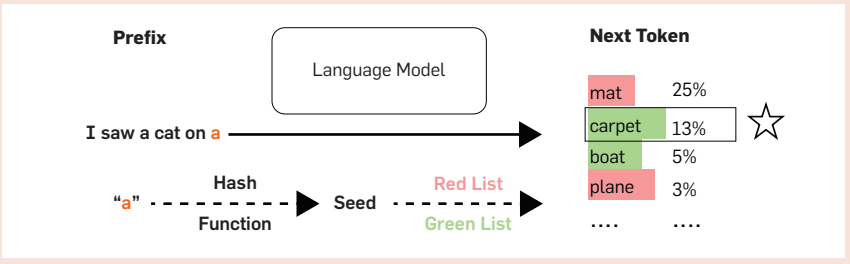
The researchers obtained responses generated by ChatGPT by presenting the same questions to the model and collecting its outputs. The evaluation results demonstrated that a RoBERTa-based detector achieved the highest performance, with F1 scores of 99.79% for paragraph-level detection and 98.43% for sentence-level detection of English text. The accompanying table presents more representative public benchmarking datasets for detecting different LLMs across various domains.

Despite numerous LLM-generated text detection studies, there is no comprehensive benchmarking dataset for performance comparison. This gap arises from the divergence in detection targets across different studies, focusing on distinct LLMs in various domains like news, question-answering, coding, and storytelling. Moreover, the rapid evolution of LLMs must be acknowledged. These models are being developed and released at an accelerating pace, with new models emerging monthly. As a result, it is increasingly challenging for researchers to keep up with these advancements and create datasets that accurately reflect each new model. Thus, the ongoing challenge in the field lies in establishing comprehensive and adaptable benchmarking datasets to accommodate the rapid influx of new LLMs.

Adaptive Attacks for Detection Systems

The previous section delineated both white-box and black-box detection methodologies. This section pivots toward adversarial perspectives, delving into potential adaptive attack strategies capable of breaching existing detection approaches. A representative work is from Sadasivan et al.[32] The authors empirically demonstrated that a paraphrasing attack could break a wide array of detectors, including both white-box and black-box approaches. This attack is premised on a straightforward yet potent assumption: given a sentence s, we denote $P(s)$ as the set of all paraphrased sentences that have similar meanings to

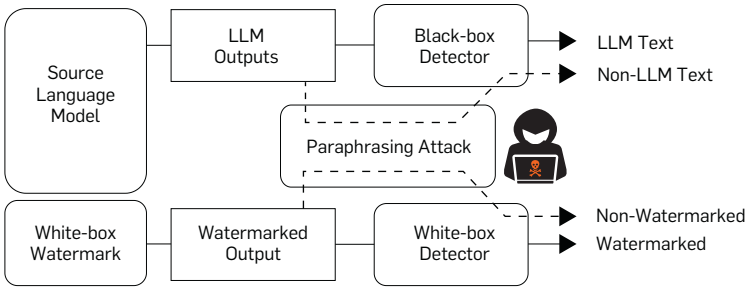
Figure 4. Illustration of inference time watermark. A random seed is generated by hashing the previously predicted token "a", splitting the whole vocabulary into "green list" and "red list." The next token "carpet" is chosen from the green list.



Representative benchmarking datasets.

Dataset	LLMs	Domain
HC3[18]	ChatGPT	Question/Answer
Neural Fake News[39]	Grover	News
TweepFake[12]	GPT2	Tweets
GPT2-Output[26]	GPT2	WebText
TURINGBENCH[37]	GPT1,2,3	News

Figure 5. Illustration of paraphrasing attack.



s. Furthermore, let $L(s)$ represent the set of sentences that the LLM could generate with a similar meaning to *s*. We assume that a detector can only identify sentences within $L(s)$. If the size of $L(s)$ is much smaller than $P(s)$, that is, $|L(s)| \ll |P(s)|$, attackers can randomly sample a sentence from $P(s)$ to evade the detector with a high probability. Based on this assumption, the author utilizes a different language model to paraphrase the output of the LLM, thereby simulating the sampling process from $P(s)$, as depicted in Figure 5. The empirical result shows that the paraphrasing attack is effective for the inference time watermark attack.[23] Utilizing the PEGASUS-based paraphrasing, the author succeeded in reducing the green list tokens from 58% to 44%. As a result, the detector accuracy drops from 97% to 80%. The attack also adversely affected black-box detection methods, causing the true positive rate of OpenAI's RoBERTa-Large-Detector to decline from 100% to roughly 80% with a practical false positive rate of 1%. Future research will inevitably encounter a proliferation of attack strategies designed to dupe detection systems. Developing robust detection systems capable of withstanding such potential attacks poses a formidable challenge to researchers.

Authors' Concerns

Limitations of black-box detection: *Bias in collected datasets.* Data collection plays a vital role in the development of black-box detectors, as these systems rely on data they are trained on to learn how to identify detection signals. However, it is important to note the data collection process can introduce biases that can negatively impact the performance and generalization of the detector.[28] These biases can take several forms. For example, many existing studies tend to focus on only one or a few specific tasks, such as question-answering or news generation, which can lead to an imbalanced distribution of topics in the data and limit the detector's ability to generalize. Additionally, human artifacts can easily be introduced during data acquisition, as seen in the study conducted by Guo et al.,[18] where the lack of style instruction in collecting LLM-generated answers led to ChatGPT producing answers with a neutral sentiment. These spurious correlations can be captured and even amplified by the detector, leading to poor generalization performance in real-world applications.

Confidence calibration. In the development of real-world detection systems, it's crucial not only to have accurate classifications but also to provide an indication of the likelihood of being incorrect. For instance, a text with a 98% probability of being generated by an LLM should be considered more likely to be machine-generated than one with a 90% probability. In other words, the predicted class probabilities should reflect its ground truth correctness likelihood. Accurate confidence scores are of paramount importance for assessing system trustworthiness, as they offer valuable information for users to establish trust in the system, particularly for neural networks whose decisions can be challenging to interpret. Although neural networks exhibit greater accuracy than traditional classification models, research on confidence score accuracy in LLM-generated text detection topics remains scarce. Therefore, it is essential to calibrate the confidence scores for black-box detection classifiers, which frequently employ neural-based models.

In our opinion, while black-box detection works at present due to detectable signals left by language models in generated text, it will gradually become less viable as language model capabilities advance and ultimately become infeasible. Considering the rapid improvement in LLM-generated text quality, the future of reliable detection tools lies in white-box watermarking detection approaches.

Limitations of white-box detection. The limitations of the white-box detection method primarily stem from two perspectives. Firstly, there exists a trade-off between the effectiveness of the watermark and the quality of the text, which applies to both post-hoc watermarks[1,21] and inference time watermarks.[32] Achieving a more reliable watermark often requires significant modifications to the original text, potentially compromising its quality. The challenge lies in optimizing the trade-off between watermark effectiveness and text quality to identify an optimal balance. Moreover, as users accumulate a growing volume of watermarked text, there is a potential for adversaries to detect and reverse engineer the watermark. Sadasivan et al.[32] emphasized that when attackers query the LLMs, they can gain knowledge about the green list words proposed in the inference time watermark.[23] This enables them to generate text that fools the detection system. To address this issue, it is essential to quantify the detectability of the watermark and explore its robustness under different query budgets. These avenues present promising directions for future research.

Lacking comprehensive evaluation metrics. Existing studies often rely on metrics such as AUC or accuracy for evaluating detection performance. However, these metrics only consider an average case and are not enough for security analysis. Consider comparing two detectors: Detector A perfectly identify of 1% of the LLM-generated text but succeeds with a random 50% chance on the rest. Detector B succeeds with 50.5% on all data. On average, two detectors have the same detection accuracy or AUC. However, detector A demonstrates exceptional potency, while detector B is practically ineffective. To know if the detector can reliably identify the LLM-generated text, researchers must consider the low false-positive rate regime (FPR) and report a detector's true-positive rate (TPR) at a low false-positive rate. This

objective of designing methods around low false-positive regimes is widely used in the computer security domain.[22] This is especially crucial for populations who produce unusual text, such as non-native speakers. Such populations might be especially at risk for false-positive, which could lead to serious consequences if these detectors are used in our education systems.

Threats from open source LLMs. Current detection methods assume the LLM is controlled by the developers and offered as a service to end-users, this one-to-many relationship is conducive to detection purposes. However, challenges arise when developers open source their models or when hackers steal them. For instance, Meta's latest LLM—LLaMA— was initially accessible by request. But just a week after accepting access requests, it was leaked online via a 4chan torrent, raising concerns about the potential surge in personalized spam and phishing attempts.[38] Once the end user gets full access to the LLM, the ability to modify the LLMs' behavior hinders black-box detection from identifying generalized language signals. Embedding watermarks in LLM outputs is one potential solution, as developers can integrate concealed watermarks into LLM outputs before making the models open source. However, it can still be defeated as users have full access to the model and can fine-tune it or change sampling strategies to erase existing watermarks. Furthermore, it is challenging to impose the requirement of embedding a traceable watermark on all open source LLMs.

With a growing number of companies and developers engaging in open source LLM projects, a future possibility emerges wherein individuals can own and customize their own language models. In this scenario, the detection of open source LLMs becomes increasingly complex and challenging. As a result, the threat emanating from open source LLMs demands careful consideration, given its potentially wide-ranging implications for the security and integrity of various sectors.

Conclusion

The detection of LLM-generated text is an expanding and dynamic field, with numerous newly developed techniques emerging continuously. This survey provides a precise categorization and in-depth examination of existing approaches to help the research community comprehend the strengths and limitations of each method. Despite the rapid advancements in LLM-generated text detection, significant challenges still must be addressed. Further progress in this field will require developing innovative solutions to overcome these challenges. C

References

1. Abdelnabi, S. and Fritz, M. Adversarial watermarking transformer: Towards tracing text provenance with data hiding. In *Proceedings of the 2021 IEEE Symp. Security and Privacy*. IEEE, 121–140.
2. Atallah, M.J. et al. Natural language watermarking: Design, analysis, and a proof-of concept implementation. *Proceeding of Information Hiding: The 4th Intern. Workshop (Pittsburgh, PA, USA, April 25–27, 2001)*. Springer, 185–200.
3. Bhatt, P. and Rios, A. Detecting bot-generated text by characterizing linguistic accommodation in human-bot interactions. *Findings of the Association for Computational Linguistics*. ACL, Bangkok, Thailand, 2021, 3235–3247.
4. Brassil, J.T., Low, S., Maxemchuk, N.F. and O'Gorman, L. Electronic marking and identification techniques to discourage document copying. *IEEE J. Selected Areas in Commun. 13*, 8 (1995), 1495–1504.
5. Brown, P.F. et al. An estimate of an upper bound for the entropy of english. *Computational Linguistics 18*, 1 (1992), 31–40.
6. Clark, E. et al. All that's 'human' is not gold: Evaluating human evaluation of generated text. In *Proceedings of the 59th Annual Meeting of the ACL and the 11th Intern. Joint Conf. on NLP*. ACL, Bangkok, Thailand, 2021, 7282–7296.
7. Crothers, E., Japkowicz, N., Viktor, H. and Branco, P. Adversarial robustness of neural-statistical features in detection of generative transformers. In *Proceedings of the 2022 Intern. Joint Conf. Neural Networks*. IEEE, Padua, Italy, 2022, 1–8.
8. Dugan, L., Ippolito, D., Kirubarajan, A. and Callison-Burch, C. Roft: A tool for evaluating human detection of machine-generated text. In *Proceedings of the 2020 Conf. Empirical Methods in NLP: System Demonstrations*. ACL Online, 2020, 189–196.
9. Dugan, L. et al. Real or fake text? Investigating human ability to detect boundaries between human-written and machine-generated text. In *Proceedings of the 37th AAAI Conf. on AI*. AAAI, Washington, D.C., USA, 2023.
10. Ede-Osifo, U. *College Instructor Put on Blast for Accusing Students of Using Chatgpt on Final Assignments*, 2023; https://nbcnews.to/3G15wg6
11. Elsen-Rooney, M. *NYC Education Department Blocks Chatgpt on School Devices, Networks*, 2023; https://bit.ly/47kmnXz
12. Fagni, T. et al. Tweepfake: About detecting deepfake tweets. *Plos One 16*, 5 (2021), e0251415.
13. Fan, A. et al. Eli5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the ACL (Florence, Italy, 2019)*, 3558–3567.
14. Floridi, L. and Chiriatti, M. GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines 30* (2020), 681–694.
15. Fröhling, L. and Zubiaga, A. Feature-based detection of automated language models: Tackling GPT-2, GPT-3 and grover. *PeerJ Computer Science 7* (2021), e443.
16. Gallé, M., Rozen, J., Kruszewski, G. and Elsahar, H. *Unsupervised and Distributional Detection of Machine-Generated Text*, 2021, arXiv:2111.02878.
17. Gehrmann, S., Strobelt, H. and Rush, A.M. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the ACL: System Demonstrations*, 2019, 111–116.
18. Guo, B. et al. How close is chatgpt to human experts? *Comparison Corpus, Evaluation, and Detection*. 2023, arXiv:2301.07597.
19. Holtzman, A. et al. *The Curious Case of Neural Text Degeneration*. 2019, 1–16, arXiv:1904.09751.
20. Ippolito, D., Duckworth, D., Callison-Burch, C. and Eck, D. Automatic detection of generated text is easiest when humans are fooled. In *Proceedings of the 58th Annual Meeting of the ACL*, 2020, 1808–1822.
21. Jalil, Z. and Mirza, A.M. A review of digital watermarking techniques for text documents. In *Proceedings of IEEE 2009 Intern. Conf. Information and Multimedia Technology*. IEEE Computer Society, Washington D.C., U.S., 230–234.
22. Kantchelian, A. et al. Better malware ground truth: Techniques for weighting anti-virus vendor labels. In *Proceedings of the 8th ACM Workshop on AI and Security*. ACM, Denver, CO, USA, 2015, 45–56.
23. Kirchenbauer, J. et al. *A Watermark for Large Language Models*. 2023, arXiv:2301.10226.
24. Liu, Y. et al. *Roberta: A Robustly Optimized BERT Pretraining Approach*. 2019, 1–13, arXiv:1907.11692.
25. Massarelli, L. et al. How decoding strategies affect the verifiability of generated text. In *Findings of the ACL: EMNLP 2020*. ACL Online, 223–235.
26. OpenAI. *Gpt-2-Output-Dataset*. 2021; https://github.com/OpenAI.openai/gpt-2-output-dataset.
27. Pagnoni, A., Graciarena, M. and Tsvetkov, Y. Threat scenarios and best practices to detect neural fake news. In *Proceedings of the 29th Intern. Conf. Computational Linguistics.(Gyeongju, Republic of Korea, 2022)*, 1233–1249.
28. Pagnoni, A., Graciarena, M. and Tsvetkov, Y. Threat scenarios and best practices to detect neural fake news. In *Proceedings of the 29th Intern. Conf. Computational Linguistics*. 2022, 1233–1249.
29. Piantadosi, S.T. Zipf's word frequency law in natural language: A critical review and future directions. *Psychonomic Bulletin Spsampsps Review 21*, (2014), 1112–1130.
30. Rodriguez, J. et al. Cross-domain detection of GPT-2-generated technical text. In *Proceedings of the 2022 Conf. North American Chapter of the ACL: Human Language Technologies. (ACL Seattle, WA, U.S., 2022)*, 1213–1233.
31. Rosenblatt, K. *Chatgpt Passes MBA Exam given by a Wharton Professor*, 2023; https://bit.ly/3G9824b
32. Sadasivan, V.S. et al. *Can AI-generated Text Be Reliably Detected?* 2023, arXiv:2303.11156.
33. Shakeel, D. and Jain, N. *Fake News Detection and Fact Verification Using Knowledge Graphs and Machine Learning*. Feb. 2021, 1–7.
34. Solaiman, I. et al. *Release Strategies and the Social Impacts of Language Models*. 2019, 1–46, arXiv:1908.09203.
35. Tillmann, C. and Ney, H. Word reordering and a dynamic programming beam search algorithm for statistical machine translation. *Computational Linguistics 29*, 1 (2003), 97–133.
36. Topkara, U., Topkara, M. and Atallah, M.J. The hiding virtues of ambiguity: Quantifiably resilient watermarking of natural language text through synonym substitutions. In *Proceedings of the 8th Workshop on Multimedia and Security*. ACM Geneva, Switzerland, 2006, 164–174.
37. Uchendu, A. et al. TURINGBENCH: A benchmark environment for turing test in the age of neural text generation. In *Findings of the ACL EMNLP*, 2021, 2001–2016.
38. Vincent, J. *Meta's Powerful AI Language Model Has Leaked Online—What Happens Now?* 2023; https://bit.ly/49I0gf7
39. Zellers, R. et al. Defending against neural fake news. *Advances in Neural Information Processing Systems 32* (2019), 9054–9065.
40. Zhong, W. et al. Neural deepfake detection with factual structure of text. In *Proceedings of the 2020 Conf. Empirical Methods in NLP*. ACL, Minneapolis, MN, 2020, 2461–2470.

Ruixiang Tang is a Ph.D. student in the Department of Computer Science, Rice University, Houston, TX, USA.

Yu-Neng Chuang is a Ph.D. student in the Department of Computer Science, Rice University, Houston, TX, USA.

Xia Hu is an associate professor in the Department of Computer Science, Rice University, Houston, TX, USA.

Watch the authors discuss this work in the exclusive *Communications* video. https://cacm.acm.org/videos/detecting-llm-generated-texts