Synthetic Text Detection in the Age of Large Language Models: Watermark vs. Automatic Detection

Adaku Uchendu
MIT Lincoln Laboratory
Lexington, MA, USA
adaku.uchendu@ll.mit.edu

Abstract

Given the ubiquitous nature of Large Language Models (LLMs) and its impressive capabilities, malicious uses of this technology to generate harmful content have been observed. Thus, to mitigate this serious security risk LLMs pose, many researchers have proposed two techniques for detecting synthetic texts generated from LLMs - watermark and automatic detection. The idea with watermarking LLMs involves infusing generated content with algorithmically-identifiable patterns during generation. This makes accurate synthetic text detection achievable with watermark detection. While, for automatic detection, the focus is on using statistical and linguistic cues to reveal authorship of texts as human or LLM. Currently, both types of synthetic text detectors achieve state-of-the-art performance, however, the better detector is still unknown. To ascertain the better detection method, we evaluate each method on their performance on both unperturbed and perturbed (i.e., adversarially manipulated texts) data. We perform a comprehensive study across six different sizes of Qwen2.5 models, six watermark techniques and detectors, two automatic detectors, three authorship obfuscation methods for different levels of syntactic changes, and two datasets of different text lengths. Our results suggest that there is no detector that consistently outperforms on all scenarios. However, we observe that the (1) automatic detectors are better for short synthetic text detection; and (2) watermark detectors perform better defending against the word-level attack implemented.

1 Introduction

Synthetic Text generation has become a ubiquitous activity with the advent of Large Language Models (LLMs) such as ChatGPT. These models are now able to generate high quality long coherent texts that look almost indistinguishable from human-written texts (Uchendu et al., 2023a; Lucas et al., 2023). While, this capability creates avenues for promising applications such as providing virtual assistants (Guan et al., 2023), software development (Jin et al., 2025), Education assistance (Chu et al., 2025) etc., the potential malicious uses make LLMs (i.e., popular generators of synthetic texts) a security risk (Uchendu et al., 2023a). This is because LLMs can be used to create authentic-looking human-like synthetic texts for malicious uses like terrorism recruitment, toxic and hate speech, mal-information, etc (Lucas et al., 2023). Thus, to combat these obvious security risks, it is imperative to accurately attribute authorship of human-written vs. synthetic texts.

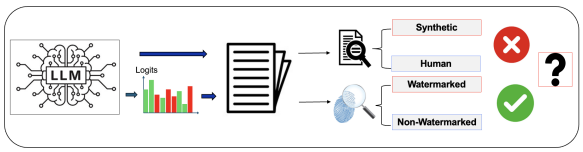


Figure 1: Watermark detection vs. Automatic detection

Therefore, in this study we explore ways to achieve accurate attribution through *synthetic text detection*. This will inform the decision for which type of synthetic text detectors needs to be deployed for a specific use case. Given the almost indistinguishable nature of recent synthetic texts, we explore two popular distinct approaches - Pre-hoc which involve *watermarking* LLMs (during generation) (Kirchenbauer et al., 2023), such that accurate detection is guaranteed due to using a forced schema during generation; and Post-hoc which involves *automatic detection* using tools that use statistical and linguistic features to determine authorship of texts (Hans et al., 2024). While both techniques have achieved state-of-the-art performances, the better technique is still unknown, thus we perform the first comprehensive study to the best of our knowledge to ascertain the better technique. See Figure 1 for an illustration of the problem - watermark vs. automatic detection. Con-

sequently, we evaluate the performance of these techniques on (1) detection of synthetic texts - both short and long texts, and (2) robustness of the detectors to authorship obfuscation techniques. Thus, we investigate four Research Questions (RQs):

- **RQ1**: What is the quality of the generated texts?
- **RQ2**: How well does Watermark detection perform at synthetic text detection?
- **RQ3**: How well does Automatic detection perform at synthetic text detection?
- **RQ4**: Which detection method (i.e., watermark vs. automatic) is more robust to Authorship obfuscation?

To answer these RQs, we evaluate on six LLMs of different sizes from the Qwen2.5 model family - 1.5B, 3B, 7B, 14B, 32B, and 72B. Next, for **RQ1**, we use both reference-free and reference-based metrics to evaluate the quality of the generated texts and the semantic and lexical alignment between watermarked and unwatermarked texts. For **RQ2**, we use six different watermarking techniques and for **RQ3**, we use two types of automatic detectors - deep learning-based and statistical-based. Lastly, for **RQ4**, we employ three semantic-preserving obfuscation techniques that perform changes at different syntactic levels - character-level, word-level, and sentence-level to the synthetic texts.

Finally, in this study, we aim to ascertain which is the better synthetic text detector - watermark vs. automatic detector and in which scenarios does it perform well.

2 Problem Definition

2.1 RQ1: Quality of Generated Texts

Before assessing the robustness of detection methods for synthetic texts, we must first evaluate the quality of these generated texts. We use the following types of metrics below:

- **Reference-based metrics:** First, we use reference-based metrics to evaluate the similarity between unwatermarked texts and watermarked texts (generated by different watermarking techniques). This will increase the trustworthiness of the watermarked texts quality, as well as detectablity.
- **Reference-free metrics:** We use reference-free metrics to evaluate the text quality of all generated texts, to ensure the we are using high quality generated texts for our experiments.

2.2 RQ2: Watermark Detection

We define watermark, in the context of synthetic text detection as a *pattern in text that is hidden to human naked eyes but algorithmically identifiable as machine-generated* (Kirchenbauer et al., 2023). By watermarking LLM-generated texts, we can confidently detect these texts, generated using the watermark schema, provided the key is known. Detection of synthetic text is becoming difficult and onerous to both detectors (Wang et al., 2024) and humans (Uchendu et al., 2023b), thus watermarking LLMs provides an avenue to perform detection confidently. Thus, we aim to answer the research question - *How well does Watermarking LLM perform in synthetic text detection?* See Table 1 for description of the six different watermark techniques we use for our evaluation. Finally, we use the MarkLLM framework[1] (Pan et al., 2024) to evaluate all the watermarkers.

2.3 RQ3: Automatic Detection

We define automatic detection, in the context of synthetic text detection as the *process of distinguishing LLM-generated texts from human-written texts using statistical and linguistic analysis*. Since, watermarking is not yet a widely adopted technique by LLM creators, we must rely on distinguishing text authors using statistical and linguistic features that reveal an authors writing style. Furthermore, unlike watermark detectors, state-of-the-art automatic detectors, aim to generalize to other LLMs, while watermark detectors can only confidently detect texts generated using its watermark schema. Thus, we aim to answer the research question - *How well does Automatic detection work for synthetic text detection?* Using two different styles of detectors - deep learning-based and statistical-based, we compare two state-of-the-art automatic detector techniques. For a fair comparison with the watermark detector, we evaluate these automatic detectors zero-shot style.

- **MAGE** (Li et al., 2024) is a deep learning-based technique which uses longformer (Beltagy et al., 2020) as its backbone to train on synthetic texts in-the-wild.
- **Binoculars** (Hans et al., 2024) is a statistical-based technique which uses FALCON (Almazrouei et al., 2023) LLM as its back-bone to calculate the perplexity of texts and uses a

[1] https://github.com/THU-BPM/MarkLLM

Technique	Description
KGW	(Kirchenbauer et al., 2023) proposes a technique to randomly create green and red list of words, such that watermarked texts contains words mostly from the green list.
UPV	(Liu et al., 2024a) proposes an Unforgeable Publicly Verifiable (UPV) watermark algorithm using two neural network to perform the watermarking and detection.
SWEET	(Lee et al., 2024) proposes a Selective WatErmarking via Entropy Thresholding (SWEET) algorithm for code generation but can be generalized to text generation as well.
EWD	(Lu et al., 2024) proposes an Entropy-based Text Watermarking Detection (EWD) to improve watermarking and detection of watermarked texts.
TS	(Huo et al., 2024) proposes a Token-Specific (TS) watermarking technique that leverages a multi-objective optimization approach for watermarking.
Unbiased	(Hu et al., 2024) proposes an unbiased watermarking technique to mitigate the trade-off between watermarking robustness and generation quality.

Table 1: Description of the six Watermark techniques we select for our study

threshold to distinguish synthetic and human-written texts.

2.4 RQ4: Authorship Obfuscation

We aim to answer the research question - *how well do both detection techniques perform on semantic-preserving adversarial perturbations?* For a more comprehensive evaluation, we use three semantic-preserving adversarial attacks which make syntactic changes in the character-level, word-level, and sentence-level of the texts. See attacks:

- **Homoglyph attack** is a character-level attack which replaces characters in texts with a different alphabet unicode (e.g., Cyrillic $\rightarrow$ Latin).
- **Misspelling attack** is a word-level attack that randomly replaces words in texts with their popular misspelling in the English language (e.g., their $\rightarrow$ thier).
- **Paraphrasing attack** is a sentence-level attack that re-writes the entire text, changing the authors style, while preserving the semantics. We use Mistral zero-shot prompt (Alperin et al., 2025), a state-of-the-art technique for paraphrasing texts.

Technique	Text
Prompt	**Provide a motivational quote about embracing change**
Unwater	To embrace change is to embrace life
KGW	A journey of a thousand miles begins with a single step.
UPV	A journey of a thousand miles begins with a single step, as does the embrace of change.
SWEET	Embrace change, for it is the only constant in life.
EWD	The future belongs to those who believe in the beauty of their dreams.-Eleanor Roosevelt
TS	A change is as good as a rest.
UNBIASED	A journey of a thousand miles begins with a single step.

Table 2: Example texts of Motivational quotes produced by different watermark techniques using Qwen2.5-3B.

3 Methodology

In order to understand how these synthetic text detectors - watermark and automatic detection perform on different model sizes, we evaluate on six model sizes in the Qwen2.5 LLM family - 1.5B, 3B, 7B, 14B, 32B, and 72B. By evaluating on models from the same family, we can mitigate for other variables such as different training data and alignment techniques that may make the models perform better or worse.

3.1 Data Description

We use two datasets to comprehensively study how these detectors perform on different text lengths - short-text and long-texts. For short-texts (i.e., $<$ 100 words), we use a motivational quotes[2] dataset that contains prompts. We sampled a smaller subset of the dataset, yielding 1066 samples. Also, since we generate unwatermarked texts, as well as watermarked texts from six watermark techniques, each LLM size has 7×1066 generations. See Table 2 for examples of the short texts, generated with Qwen2.5-3B for all six watermark techniques. Next, we performed analysis on longer texts (i.e., $>$ 450 words) using a fiction dataset[3]. We also sampled a subset of the dataset, yielding 1035 samples. Also, since we generate unwatermarked texts, as well as watermarked texts from six watermark techniques, each LLM size has 7×1035 generations. See Table 5 in Appendix, for examples of the long texts, generated with Qwen2.5-3B for all six watermark techniques. This fiction dataset contains instructions for the LLM to use while generating texts. Additionally, see both Tables 6 and 7 in Appendix for the average word count and sentence count for the generated short and long texts, respectively.

[2] https://huggingface.co/datasets/asuender/motivational-quotes

[3] https://huggingface.co/datasets/Dans-DiscountModels/RUCAIBox-Story-Generation-Alpaca

3.2 Evaluation Metrics

3.2.1 Linguistic and Statistical Metrics.

We employ linguistic and statistical metrics to quantify the quality of generated texts so that our detection results can be meaningful. Thus, we employ both reference-based (i.e., requiring text pairs) and reference-free (i.e., requiring a piece of text) metrics. See metrics below:

Reference-based Metrics

- *BERTScore*: Measures the semantic and lexical similarity of text pairs. The score range is [0,1], and close to 1 is ideal (Zhang et al., 2019).
- *Cosine Similarity*: Measures the lexical similarity of text pairs. The score range is [0,1], and close to 1 is ideal.
- *Levenshtein distance*: Measures the edit distance between two texts - T1 and T2. It calculates the number of edits required to make T2 = T1. We calculate the word-based edit distance. There is no range for the score and close to 0 is ideal.

Reference-free Metrics.

- *Entropy*: Measures the information contained in text. Typically high entropy means informative, however since the metric calculates high information by measuring surprisal. In our case high entropy means lower quality of texts because generated texts typically follow a specific distribution, maintaining low surprisal.
- *SMOG*: Estimates the years of education of a writer (Mc Laughlin, 1969).
- *Flesch Kincaid Grade*: Measures how easy it is to read a text. Scores are in range [0, 100], where lower score below 30 indicates difficulty in reading (which indicates college-level writing) (Flesch, 1948).

3.2.2 Performance Metrics

To calculate the performance of the detectors both on unperturbed and perturbed data, we only use Accuracy. More specifically we use per-class accuracy, meaning that we calculate the performance of the model at detecting a specific class only.

4 Results

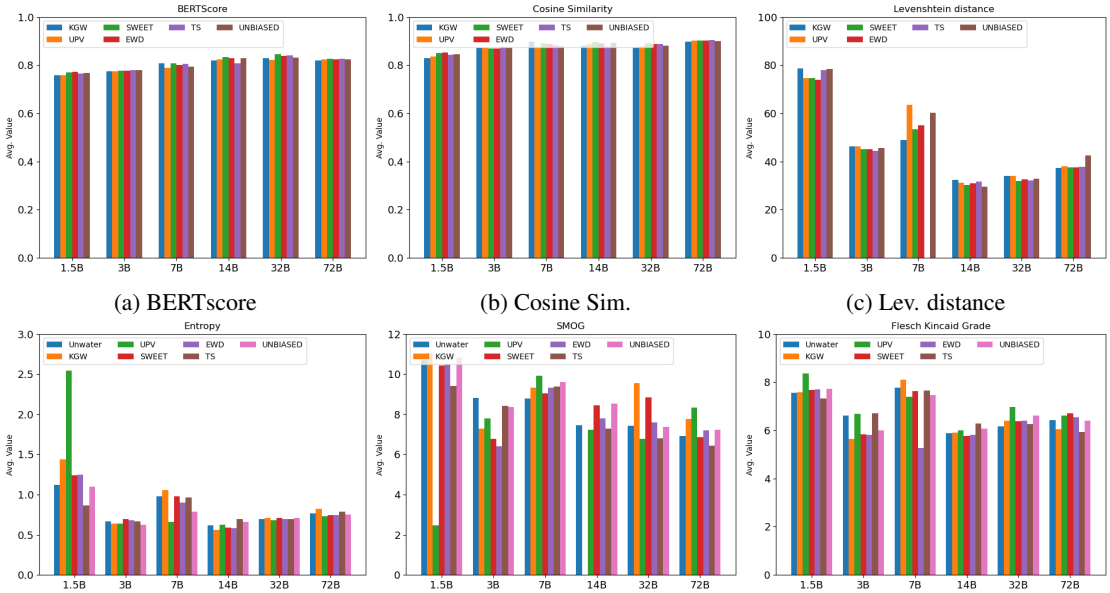
4.1 RQ1: Quality of Generated Texts

See Figures 2 & 3 for the plots of these metrics, for short and long synthetic texts. First, we measure the lexical and semantic similarity between watermarked and unwatermarked texts to ascertain the quality of watermarked texts. Thus, we use the reference-based metrics - BERTScore, cosine similarity, and Levenshtein distance. We observe consistent high scores (i.e., > 0.7) for both BERTScore and cosine similarity, suggesting high semantic and lexical similarity on all six models and six watermarkers for both short and long texts. Next, we observe some inconsistencies between the model sizes for Levenshtein distance - first, for short texts, the larger models - 14B, 32B, and 72B have lower Levenshtein distance, with 1.5B, exceeding all models by a large margin. For long texts, the pattern is different, such that the edit distances are closer together, although 1.5B and 14B has a slightly larger edit distance than the other model sizes.

Second, we measure the writing quality for each generated texts. We use the reference-free metrics - Entropy, SMOG, and Flesch Kincaid grade. For short texts, we observe disparities in all these metrics. Entropy is mostly similar, with a very large difference in UPV for the 1.5B model. SMOG experiences some differences that are harder to interpret but one that stands out is that UPV for 1.5B has a lower score than other watermarkers, but has a highest score for 7B and 72B. Next, the Flesch Kincaid grade is lower for the larger models - 14B, 32B, and 72B. Finally, for long texts, we observe more consistency in Entropy and SMOG, with UPV, achieving the highest score for all models. Lastly, for the Flesch Kincaid grade, UPV has the highest score for all but the 14B model, where it achieves the second highest.

4.2 RQ2: Watermark Detection

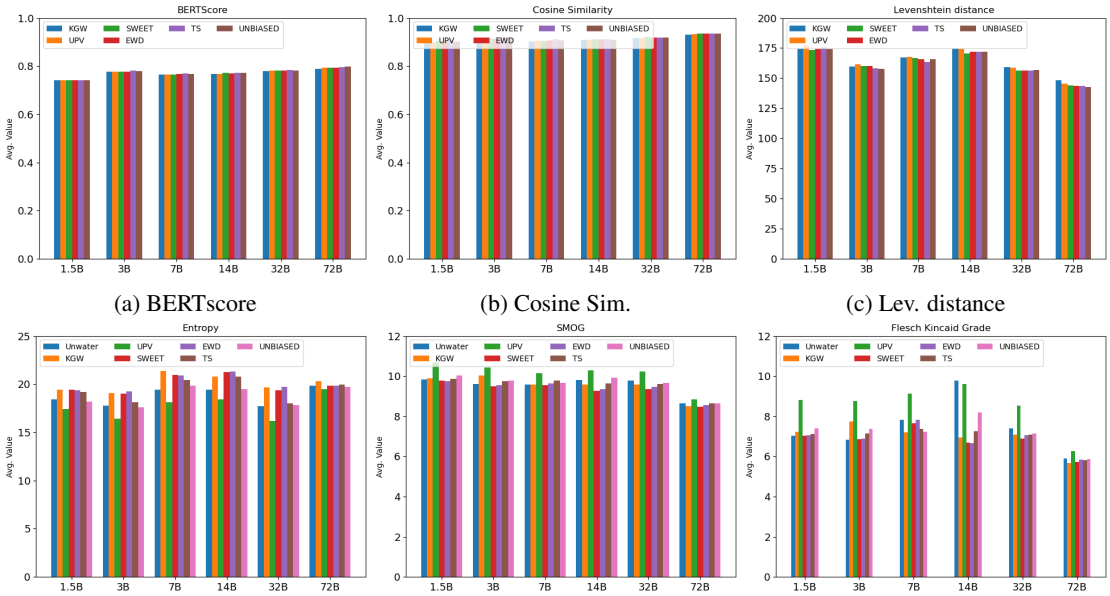
To answer the question of *is this text, watermarked?*, we perform a binary classification of watermarked and unwatermarked texts. See Figure 4. We observe that the detector performs better on detecting watermarked texts for longer generations than short. For short texts, the detectors performs best on detecting only unwatermarked texts accurately, achieving 100% accuracy and underperform by large margins, the greatest being 14B, 32B, and 72B on detecting watermarked texts. We observe a less than 10% accuracy for all watermark detectors, except UPV on 1.5B and much lower on 3B and 7B. UPV has the best performing watermark detector, achieving about 60% accuracy on 1.5B and lower for the other models, but much higher than



(a) BERTscore (b) Cosine Sim. (c) Lev. distance

(d) Entropy (e) Smog (f) Flesch K. Grade

Figure 2: Short-text Reference-based (above) Reference-free (below) metrics for **Short** texts



(a) BERTscore (b) Cosine Sim. (c) Lev. distance

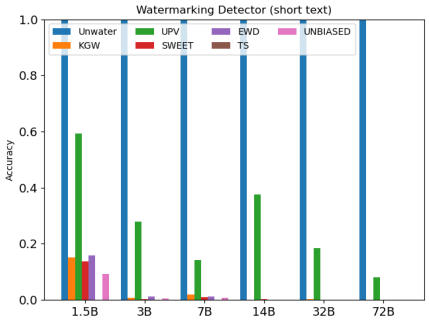
(d) Entropy (e) Smog (f) Flesch K. Grade

Figure 3: Reference-based (BERTScore, Cosine Similarity, Levenshtein distance) and Reference-free (Entropy, Smog, Flesch Kincaid Grade) metrics for **Long** texts

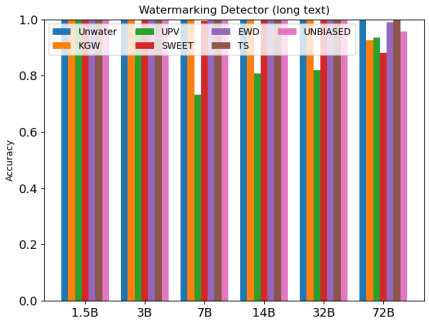
the other watermark detectors. For long texts, we observe over a 90% accuracy in both the watermark and unwatermark detection, except a slightly less performance of UPV on 7B, 14B, and 32B.

4.3 RQ3: Automatic Detection

To answer the question - *is this text, synthetic?*, we perform a binary classification using state-of-the-art zeroshot automatic synthetic text detectors. Thus, we use Binoculars, a statistical-based technique, and MAGE, a deep learning-based technique. See results in Figure 5. We observe that Binoculars achieves between 55-80% accuracy for short texts, and 86-98% accuracy for long texts. In addition, MAGE achieves between 86-98% accuracy for short texts, and about 100% accuracy for long texts. The detection accuracy are highest on the 72B model for both short and long texts. Additionally, for both short and long texts, unwatermarked and watermarked perform similarly in



(a) Short-text



(b) Long-text

Figure 4: Watermark Detector results for short and long texts

detectability.

	Watermark detector			Automatic detector		
	Homg.	Miss.	Para.	Homg.	Miss.	Para.
Unwater						
KGW						
UPV						
SWEET						
EWD						
TS						
UNBIASED						

Table 3: Detection performance across watermark and automatic detectors on Obfuscated texts. The columns are the three attack techniques - Homg.: Homoglyph, Miss.: Misspellings, and Para. Paraphrasing. Green = strong detection (i.e., $> 50\%$), red = under performance (i.e., $< 50\%$), light green/red = partial performance.

4.4 RQ4: Authorship Obfuscation

We employ semantic-preserving authorship obfuscation techniques that adversarially make changes to the texts on different levels - character-level, word-level, and sentence-level. See examples of perturbed texts in Table 8 in Appendix and Table 3 for a summary of the obfuscation results. For the detailed results, see Figure 6 in Appendix.

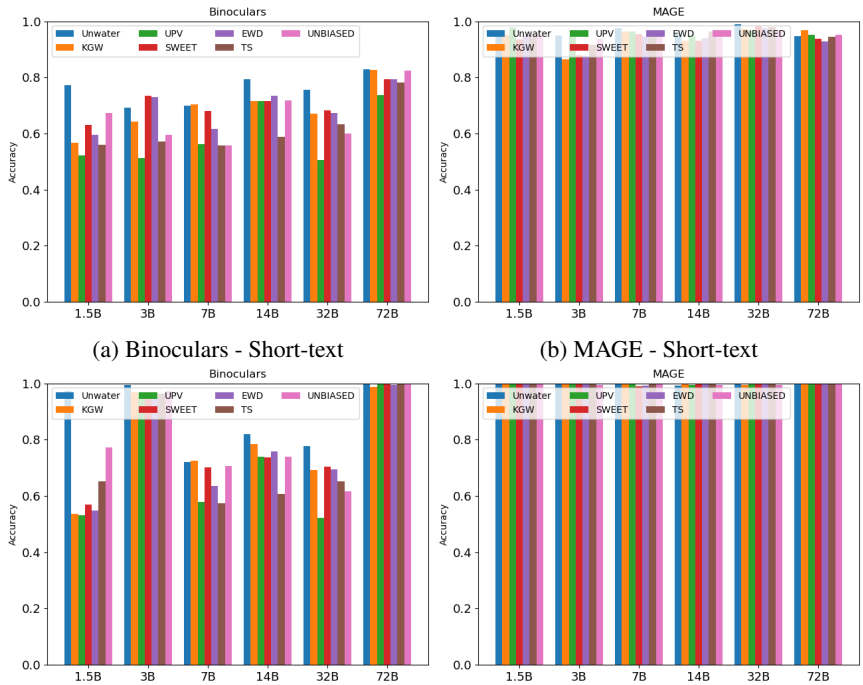
For the character-level attack, we use Homoglyph attack and observe a significant dip in performance for all detectors, with only MAGE achieving the highest accuracy (i.e., 10-40%), while the watermark detector maintains high accuracy for only unwatermarked texts and TS watermarked texts. Next, for the word-level attack, we use Misspellings attack and observe that the watermark detectors - KGW, EWD, SWEET, and TS are the most robust to this attack as they maintain high accuracy for almost all model sizes, except 72B. TS has the best watermark detector on this attack, while for the automatic detectors, Binoculars and MAGE underperform on this attack. Finally, for the sentence-level attack, we prompt Mistral to paraphrase the texts, and observe that MAGE is the most robust to this attack, maintaining high accuracy on all generated text types and model sizes. The watermark detector, performs similarly on this attack, as it did on the homoglyph attack.

5 Discussion

Watermark detection works better for unperturbed long texts. We observe from Figure 4 that watermark detection underperforms on short texts, barely achieving up to 10% accuracy in detecting its on generated texts. But achieves about 100% accuracy on detecting long texts. We believe that this may occur because with longer texts, the watermarker has more content to infuse with its signature, which increases the likelihood for being algorithmically-identifiable. Therefore, the results suggest that watermark techniques are currently suitable for longer texts and so should only be applied in long text contexts, such as essay writing.

Automatic detection works well on both unperturbed short and long texts. We observe from Figures 5 that automatic detectors, Binoculars and MAGE perform much better than watermark detectors in detecting short texts. Although, MAGE performs the best, achieving over 90% accuracy. Next, for the long texts context, Binoculars performed decently, while MAGE maintained the same high performance, suggesting that MAGE performs well on both contexts. Thus, MAGE is the better automatic detector, achieving over 90% accuracy for all unwatermarked and watermarked texts. This could suggest that MAGE, a deep learning model that trains on synthetic text in the wild, forces the model to generalize better than Binoculars, a statistical technique.

The character-level - Homoglyph attack is the most robust attack. According to Table 3 both watermark and automatic detectors are highly sus-



(a) Binoculars - Short-text

(b) MAGE - Short-text

(c) Binoculars - long-text

(d) MAGE - long-text

Figure 5: Automatic detection results using Binoculars and MAGE for both **short** (above) and **long** (below) texts

ceptible to this attack. For watermark detection, only the unwatermarked and TS watermarked texts are robust to this attack. However, from Table 4, we understand that TS is robust to this attack because it typically detects both watermarked and unwatermarked texts as watermarked. Also, for automatic detection, this attack was able to successfully obfuscate authorship, causing the detectors to misclassify the synthetic texts as human-written. Thus, using this simple attack, we reveal how brittle both detector types are to this type of character-level attack.

Watermark and Automatic Detectors are robust to different types of attacks. While, both detectors are susceptible to the character-level attack, we observe from Table 3 that these detectors are robust to different attacks. The watermark detectors are robust to the word-level attack - Misspellings attack, were KGW, SWEET, EWD, and TS maintained high accuracies, while automatic detectors underperformed. However, only one automatic detector - MAGE is robust to the sentence-level attack - Paraphrasing, maintaining accuracy above 90%. This suggest that each synthetic text detector has utility as they are robust to different attacks.

No one size fits all for all model sizes and detectors. Based on the performance of both types of synthetic text detectors on unperturbed and perturbed synthetic texts, we conclude that there is no best detector for all scenarios. In addition, we found that no model size was more or less easily detectable by all detectors. Therefore, detector users must evaluate the constraints of their scenarios before choosing a suitable detector for their use case. For instance, while the homoglyph attack was robust, it is also easy to defend against using a simple text editing tool. Therefore, detectors vulnerable to this attack can be deployed in scenarios where text editing tools are automatically embedded.

6 Conclusion

We set out to answer the question of which is the better synthetic text detector - *watermark or automatic detector*? To comprehensively study this problem, we evaluate across different variables. The results suggest that since no detector consistently outperformed on all constraints, the answer as to which is the better detector is more nuanced. Thus, since the detectors are sensitive to different text lengths and obfuscation strategies, these need to be considered before selecting and deploying a synthetic text detector.

Limitations

- **Data:** Although, we used two different datasets of different text lengths, we believe that our study could have also benefited from

many text lengths as well as domains. Especially, since our results suggest that the watermark detector is sensitive to text lengths.

- **LLM architecture:** We use six different model sizes in the same model family to mitigate for results being affected by different training strategies and dataset. However, our study could have benefited from comparing to other LLM families to observe consistent or inconsistent themes. For instance, comparing to different models sizes in the LLaMA-3.3 family.

- **Different types of Watermarkers:** Although, we compared six watermarkers, where most techniques are introduced during logit generation. It could be beneficial to select several watermark techniques across the three different strategies - during logit generation, token sampling, and during LLM training. By comparing these strategies, we can ascertain the best types of watermarkers.

- **More Obfuscators under each type:** Our study could have also benefited from selecting multiple obfuscation techniques per type, such that each of the three types of obfuscation will have, perhaps three techniques. This will help us draw stronger conclusion as to whether watermark detectors are robust to all word-level attacks or specifically, the misspellings attack.

- **Dialects and Languages:** We only focus on standard English for our study, however, it will be beneficial to observe the performance of these detectors and obfuscators on other English dialects. Furthermore, we can explore different languages and their variants as well.

Ethical Statement

The proliferation of LLMs, has led to increased deployment of cyber crimes augmented by LLMs. For this study, we are specifically motivated by malicious text generations using LLMs. To mitigate this problem, many researchers have proposed techniques that largely fall under two methods - watermark detection and automatic detection. While, both perform well on synthetic text detection, it is still uncertain which detection type is better. Therefore, we perform a comprehensive study across several variables to answer this question. While, we understand that our results could be leveraged by malicious actors to improve their attack designs and vectors, we believe that benefits outweighs the risks as we found novel results that could benefit the deployment of these detectors. Finally, we do not build new detectors but evaluate the robustness of existing ones.

References

George Alexandru Adam, Alexander Cui, Edwin Thomas, Emily Napier, Nazar Shmatko, Jacob Schnell, Jacob Junqi Tian, Alekhya Dronavalli, Edward Tian, and Dongwon Lee. 2026. Gptzero: Robust detection of llm-generated texts. *Preprint*, arXiv:2602.13042.

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, and 1 others. 2023. The falcon series of open language models. *arXiv preprint arXiv:2311.16867*.

Kenneth Alperin, Rohan Leekha, Adaku Uchendu, Trang Nguyen, Srilakshmi Medarametla, Carlos Levya Capote, Seth Aycock, and Charlie Dagli. 2025. Masks and mimicry: Strategic obfuscation and impersonation attacks on authorship verification. In *Proceedings of the 5th International Conference on Natural Language Processing for Digital Humanities*, pages 102–116, Albuquerque, USA. Association for Computational Linguistics.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Shuyang Cai and Wanyun Cui. 2023. Evade chatgpt detectors via a single space. *Preprint*, arXiv:2307.02599.

Miranda Christ, Sam Gunn, and Or Zamir. 2024. Undetectable watermarks for language models. In *The Thirty Seventh Annual Conference on Learning Theory*, pages 1125–1139. PMLR.

Zhendong Chu, Shen Wang, Jian Xie, Tinghui Zhu, Yibo Yan, Jingheng Ye, Aoxiao Zhong, Xuming Hu, Jing Liang, Philip S. Yu, and Qingsong Wen. 2025. LLM agents for education: Advances and applications. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 13782–13810, Suzhou, China. Association for Computational Linguistics.

Evan Crothers, Nathalie Japkowicz, Herna Lydia Viktor, and Paula Branco. 2022. Adversarial robustness of neural-statistical features in detection of generative transformers. *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8.

Rudolph Flesch. 1948. A new readability yardstick. *Journal of applied psychology*, 32(3):221.

Chenchen Gu, Xiang Lisa Li, Percy Liang, and Tatsunori Hashimoto. 2024. On the learnability of watermarks for language models. In *The Twelfth International Conference on Learning Representations*.

Yanchu Guan, Dong Wang, Zhixuan Chu, Shiyu Wang, Feiyue Ni, Ruihua Song, Longfei Li, Jinjie Gu, and Chenyi Zhuang. 2023. Intelligent virtual assistants with llm-based process automation. *Preprint*, arXiv:2312.06677.

William Guo, Adaku Uchendu, and Ana Smith. 2025. Signature vs. substance: Evaluating the balance of adversarial resistance and linguistic quality in watermarking large language models. *Preprint*, arXiv:2511.13722.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2024. Spotting llms with binoculars: zero-shot detection of machine-generated text. In *Proceedings of the 41st International Conference on Machine Learning*, pages 17519–17537.

Zhiwei He, Binglin Zhou, Hongkun Hao, Aiwei Liu, Xing Wang, Zhaopeng Tu, Zhuosheng Zhang, and Rui Wang. 2024. Can watermarks survive translation? on the cross-lingual consistency of text watermark for large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4115–4129, Bangkok, Thailand. Association for Computational Linguistics.

Zhengmian Hu, Lichang Chen, Xidong Wu, Yihan Wu, Hongyang Zhang, and Heng Huang. 2024. Unbiased watermark for large language models. In *The Twelfth International Conference on Learning Representations*.

Mingjia Huo, Sai Ashish Somayajula, Youwei Liang, Ruisi Zhang, Farinaz Koushanfar, and Pengtao Xie. 2024. Token-specific watermarking with enhanced detectability and semantic coherence for large language models. In *Proceedings of the 41st International Conference on Machine Learning*, pages 20746–20767.

Haolin Jin, Linghan Huang, Haipeng Cai, Jun Yan, Bo Li, and Huaming Chen. 2025. From llms to llm-based agents for software engineering: A survey of current, challenges and future. *Preprint*, arXiv:2408.02479.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. In *International Conference on Machine Learning*, pages 17061–17084. PMLR.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2023. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *ArXiv*, abs/2303.13408.

Lucio La Cava, Davide Costa, Andrea Tagarelli, and 1 others. 2024. Is contrasting all you need? contrastive learning for the detection and attribution of ai-generated text. In *27th European Conference on Artificial Intelligence, ECAI 2024*. IOS Press BV.

Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. 2024. Who wrote this code? watermarking for code generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4890–4911.

Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. 2020. BERT-ATTACK: Adversarial attack against BERT using BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6193–6202, Online. Association for Computational Linguistics.

Yafu Li, Qintong Li, Leyang Cui, Wei Bi, Zhilin Wang, Longyue Wang, Linyi Yang, Shuming Shi, and Yue Zhang. 2024. MAGE: Machine-generated text detection in the wild. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 36–53, Bangkok, Thailand. Association for Computational Linguistics.

Jiacheng Liang, Zian Wang, Spencer Hong, Shouling Ji, and Ting Wang. 2025. Watermark under fire: A robustness evaluation of LLM watermarking. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 21050–21074, Suzhou, China. Association for Computational Linguistics.

Aiwei Liu, Leyi Pan, Xuming Hu, Shuang Li, Lijie Wen, Irwin King, and Philip S Yu. 2024a. An unforgeable publicly verifiable watermark for large language models. In *ICLR*.

Aiwei Liu, Leyi Pan, Yijian Lu, Jingjing Li, Xuming Hu, Xi Zhang, Lijie Wen, Irwin King, Hui Xiong, and Philip Yu. 2024b. A survey of text watermarking in the era of large language models. *ACM Computing Surveys*, 57(2):1–36.

Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. 2024. An entropy-based text watermarking detection method. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11724–11735, Bangkok, Thailand. Association for Computational Linguistics.

Jason Lucas, Adaku Uchendu, Michiharu Yamashita, Jooyoung Lee, Shaurya Rohatgi, and Dongwon Lee. 2023. Fighting fire with fire: The dual role of LLMs in crafting and detecting elusive disinformation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14279–14305, Singapore. Association for Computational Linguistics.

Dominik Macko, Robert Moro, Adaku Uchendu, Jason Lucas, Michiharu Yamashita, Matúš Pikuliak, Ivan Srba, Thai Le, Dongwon Lee, Jakub Simko, and 1 others. 2023. Multitude: Large-scale multilingual machine-generated text detection benchmark. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9960–9987.

Dominik Macko, Robert Moro, Adaku Uchendu, Ivan Srba, Jason S Lucas, Michiharu Yamashita, Nafis Irtiza Tripto, Dongwon Lee, Jakub Simko, and Maria Bielikova. 2024. Authorship obfuscation in multilingual machine-generated text detection. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 6348–6368, Miami, Florida, USA. Association for Computational Linguistics.

G Harry Mc Laughlin. 1969. Smog grading-a new readability formula. *Journal of reading*, 12(8):639–646.

Chidimma Opara. 2025. Distinguishing ai-generated and human-written text through psycholinguistic analysis. In *International Conference on Artificial Intelligence in Education*, pages 212–219. Springer.

Leyi Pan, Aiwei Liu, Zhiwei He, Zitian Gao, Xuandong Zhao, Yijian Lu, Binglin Zhou, Shuliang Liu, Xuming Hu, Lijie Wen, Irwin King, and Philip S. Yu. 2024. MarkLLM: An open-source toolkit for LLM watermarking. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 61–71, Miami, Florida, USA. Association for Computational Linguistics.

Adaku Uchendu, Thai Le, and Dongwon Lee. 2023a. Attribution and obfuscation of neural text authorship: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 25(1):1–18.

Adaku Uchendu, Thai Le, and Dongwon Lee. 2024. Topformer: Topology-aware authorship attribution of deepfake texts with diverse writing styles. In *27th European Conference on Artificial Intelligence, ECAI 2024*, pages 1446–1454. IOS Press BV.

Adaku Uchendu, Thai Le, Kai Shu, and Dongwon Lee. 2020. Authorship attribution for neural text generation. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 8384–8395.

Adaku Uchendu, Jooyoung Lee, Hua Shen, Thai Le, Dongwon Lee, and 1 others. 2023b. Does human collaboration enhance the accuracy of identifying llm-generated deepfake texts? In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 11, pages 163–174.

Saranya Venkatraman, Adaku Uchendu, and Dongwon Lee. 2024. Gpt-who: An information density-based machine-generated text detector. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 103–115.

Yuxia Wang, Jonibek Mansurov, Petar Ivanov, Jinyan Su, Artem Shelmanov, Akim Tsvigun, Chenxi Whitehouse, Osama Mohammed Afzal, Tarek Mahmoud, Toru Sasaki, Thomas Arnold, Alham Aji, Nizar Habash, Iryna Gurevych, and Preslav Nakov. 2024. M4: Multi-generator, multi-domain, and multilingual black-box machine-generated text detection. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1369–1407, St. Julian's, Malta. Association for Computational Linguistics.

Max Wolff and Stuart Wolff. 2020. Attacking neural text detectors. *arXiv preprint arXiv:2002.11768*.

Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. 2025. A survey on llm-generated text detection: Necessity, methods, and future directions. *Computational Linguistics*, 51(1):275–338.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.

A Related Work

A.1 Automatic detection for Synthetic Texts

Since the advent of current LLMs, we now have to grapple with the authenticity of information found online. To mitigate this issue, researchers have proposed synthetic text detection (Uchendu et al., 2023a; Wu et al., 2025) to ascertain authenticity by performing authorship attribution of texts. There are two ways to perform this task - binary classification (Macko et al., 2023; Wang et al., 2024), where the labels are human & machine; and multi-class classification (Uchendu et al., 2020; La Cava et al., 2024), where the labels are human and all LLM generators in the dataset. Binary classification is the more popular approach (Uchendu et al., 2023a). Furthermore, these techniques can be divided into deep learning-based (Li et al., 2024), statistical-based (Venkatraman et al., 2024), stylometric-based (Opara, 2025), Ensemble-based (Uchendu et al., 2024), and more recently API-accessible techniques such as GPTZero (Adam et al., 2026)[4]. Each technique has its pros and cons, and none consistently outperforms all others across every scenario (Macko et al., 2023).

A.2 Watermarking LLMs

Since, LLM-generated texts are becoming almost indistinguishable from human-written texts, researchers have proposed watermarking these generated texts during generation, such that if the watermark key is known, accurate synthetic text detection can be achieved. Thus, according to (Liu et al., 2024b), there are different watermark techniques based on when it is introduced during training - during logit generation (Kirchenbauer et al., 2023; Lee et al., 2024), during token sampling (Christ et al., 2024), and during LLM training (Gu et al., 2024). However, the more popular approach is the watermarking during logit generation, thus we study this approach in more detail.

A.3 Authorship Obfuscation of Synthetic Texts

Building an accurate synthetic text detector is no longer enough, we must also ascertain how robust they are to authorship obfuscation techniques. These techniques aim to adversarially perturb the texts, such that semantics are preserved but the author's writing style is stripped (Uchendu et al., 2023a). As this is a young field, there are many categories (Crothers et al., 2022), however, for this study, we aligned more with categorization for making syntactic changes at different levels of texts - character-level, word-level, and sentence-level attacks. Thus, for character-level attacks, there are homoglyph attack (Wolff and Wolff, 2020), single space attack (Cai and Cui, 2023), etc.; for word-level, there are misspellings (Wolff and Wolff, 2020), synonym swapping (Li et al., 2020), etc.; and for sentence-level, there are paraphrasing (Krishna et al., 2023), and backtranslation (He et al., 2024).

Additionally, many researchers have evaluated how well these synthetic texts detectors - watermark (Guo et al., 2025; Liang et al., 2025) and automatic (Macko et al., 2024) perform on obfuscated synthetic texts. Finally, since the better technique is still unknown, we comprehensively compared the two techniques to answer the question of which technique is better.

B Methodology

B.1 Data Creation

Using the MarkLLM framework (Pan et al., 2024), we prompt each of the six Qwen2.5 model sizes to generate unwatermarked texts and watermarked texts generated by six different techniques - KGW, UPV, SWEET, EWD, TS, and UNBIASED. For short-texts (i.e., $<$ 100 words), we use a motivational quotes dataset[5] that contains prompts. We sampled a smaller subset of the dataset, yielding 1066 samples. Additionally, for longer texts (i.e., $>$ 450 words), we use a fiction writing dataset[6]. We also sampled a subset of the dataset, yielding 1035 samples.

PROMPT FOR GENERATING MOTIVATIONAL QUOTES (I.E., SHORT TEXTS). *Using this prompt - {prompt_from_data} generate a {motivational_quote}. Do not include emojis .*

PROMPT FOR GENERATING FICTIONAL STORY (I.E., LONG TEXTS). *Using this instruction - {instruction_from_data} generate a {fictional_story} about {fictional_topic}. Do not include emojis .*

Since the original datasets for both short and long texts, contained prompts and instructions for

[4] https://gptzero.me/

[5] https://huggingface.co/datasets/asuender/motivational-quotes

[6] https://huggingface.co/datasets/Dans-DiscountModels/RUCAIBox-Story-Generation-Alpaca

the LLMs to follow, we use the following prompt structures above for generation. Additionally, we use the default hyperparameters recommended by the MarkLLM framework generation, but change the min and max length of texts to fit with text length. For short texts, we used the default, but for long texts we set min and max lengths to 500 and 700, respectively.

Next. for short texts we generated 1066 samples for each method. Since each model size had unwatermarked generations and six watermarked generations, this yielded $7 \times 1066 = 7462$ generations per LLM. Thus, for the six LLMs, we have $6 \times 7 \times 1066 = 44,772$ generations for our short-text dataset. See Table 2 for examples of the short texts, generated with Qwen2.5-3B for all six watermark techniques. Additionally, for long texts, we generated 1035 samples for each method. Similarly, since each model size had unwatermarked generations and six watermarked generations, this yielded $7 \times 1035 = 7245$ generations per LLM. Thus, for the six LLMs, we have $6 \times 7 \times 1035 = 43,470$ generations for our long-text dataset. See Table 5 for examples of the long texts, generated with Qwen2.5-3B for all six watermark techniques.

Finally, see Tables 6 and 7 (in Appendix) for the average word count and sentence count for the generated short and long texts, respectively.

	3B	7B	14B	AVG
KGW	0.9981	0.9961	0.9903	0.9948
UPV	1	0.9971	1	0.999
SWEET	1	1	1	1
EWD	1	1	1	1
TS	0	0	0	0
UNBIASED	1	1	1	1
AVG	0.833	0.8322	0.8317	

Table 4: Results on the evaluation for assessing how Watermark techniques perform in detecting Unwatermarked texts for Qwen2.5 models - 3B, 7B and 14B. Based on these results, we select UNBIASED watermark detector of all analysis.

B.2 Selection of Unwatermarked text detector

Each watermark detector is trained to be able to detect its synthetic texts watermarked with its method. Therefore, it should also be able to detect texts, that do not use its own watermark schema. However, we observe that watermark detectors do not perform similarly on the task of accurately classifying an unwatermarked texts. Thus, we performed a small analysis on each watermark detector's performance on detecting unwatermarked texts (for long texts). See Table 4 for results. Finally, since there are several accurate detectors - SWEET, EWD, and UNBIASED, we select one of them - the **UNBIASED** watermark detector for the detecting the unwatermarked texts, through out this study.

C Authorship Obfuscation of Synthetic Texts

We use three semantic-preserving obfuscation techniques to make syntactic changes to our generated texts to observe how robust they are to these realistic attacks. Thus, we employed attacks to makes changes to the texts are the character-level (i.e., Homoglyph attack), word-level (i.e., Misspellings attack), and sentence-level (i.e., Paraphrasing attack). See examples of these syntactic changes in Table 8. Additionally, see Figure 6 for the accuracy of each detector performance on the obfuscated generated texts for all the model sizes.

Technique	Text
Prompt	**Produce a short story that evolves from the given idea text**
Unwater	Once upon a time, in a small town nestled between rolling hills and dense forests, there lived a young man named Alex. Despite his fear of the unknown, Alex had always dreamed of embarking on a new adventure. ...
KGW	In a small town nestled among rolling hills, Alex stood nervously outside the trucking school. His heart pounded with fear as he contemplated his future. The prospect of starting anew in a different career loomed large. ...
UPV	In a small town nestled between rolling hills, Alex had always felt like he was treading water, unsure of what path to take in his life. The weight of his indecision was heavy, like carrying an invisible burden that threatened to crush him. ...
SWEET	In the heart of Texas, there lived a man named Alex, who felt an inexplicable pull towards the open road and its unpredictable adventures. He had been living a sedentary life in his hometown, surrounded by the familiar faces of his family and friends. ...
EWD	At first, it seemed too risky. My fear made me hesitate, and when I tried to take the CDL test, I failed. It felt like the world was closing in on me. ...
TS	Once upon a time, in a small town nestled between rolling hills and dense forests, I found myself at a crossroads. The routine of daily life had become monotonous, and despite my initial reluctance, I was compelled to venture out into the unknown. ...
UNBIASED	Once upon a time, in a small town nestled between rolling hills and dense forests, there lived a young man named Alex. ...

Table 5: Example texts of Fictional stories produced under different watermark techniques using Qwen2.5-3B.

LLMs	Unwater	KGW	UPV	SWEET	EWD	TS	Unbiased
Qwen2.5-1.5B	**33.05** (47.05)	**44.42** (70)	**76.36** (92.03)	**37.86** (56.95)	**38.12** (55.98)	25.3 (32.88)	**32.64** (48.53)
Qwen2.5-3B	17.35 (6.402)	16.59 (6.515)	17.5 (5.44)	17.42 (6.324)	17.19 (6.758)	18.15 (6.15)	16.88 (6.294)
Qwen2.5-7B	25.03 (25.03)	29.13 (16.96)	17.04 (9.744)	26.07 (19.34)	24.28 (16.5)	**26.31** (**28.27**)	20.58 (13.86)
Qwen2.5-14B	14.61 (4.471)	13.56 (4.344)	14.59 (4.373)	14.1 (4.124)	14.15 (4.11)	16.08 (5.381)	15.22 (5.155)
Qwen2.5-32B	15.02 (4.732)	16.28 (10.71)	16.27 (5.138)	15.46 (4.56)	15.34 (4.66)	15.14 (4.798)	16.2 (5.048)
Qwen2.5-72B	17.22 (4.994)	18.81 (7.343)	16.8 (4.704)	17.05 (4.682)	17.16 (4.728)	17.55 (4.665)	17.01 (5.119)
LLMs	**Unwater**	**KGW**	**UPV**	**SWEET**	**EWD**	**TS**	**Unbiased**
Qwen2.5-1.5B	**516.9** (85)	571.6 (46.09)	497 (87.8)	569.5 (43.95)	568.2 (51.86)	560.7 (62.36)	523.2 (80.84)
Qwen2.5-3B	467.5 (81.5)	528.3 (81.97)	451.6 (82.21)	519.7 (83.61)	527.2 (80.54)	494.3 (87.17)	474.8 (84.68)
Qwen2.5-7B	506.6 (91.26)	562.4 (51.3)	485.8 (86.38)	558.7 (65.82)	561.5 (64.15)	530 (77.35)	513.3 (86.17)
Qwen2.5-14B	537.6 (105.2)	**573** (42.59)	**507.4** (96.67)	**571.4** (45.58)	**575.9** (34.03)	564.7 (63.15)	**542.7** (94.08)
Qwen2.5-32B	465.4 (89.56)	523.4 (74.39)	433.4 (73.75)	504.8 (83.27)	518 (78.95)	473.3 (83.27)	465.5 (82.32)
Qwen2.5-72B	470 (77.84)	494.4 (76.82)	462 (76.71)	465.4 (75.71)	472.1 (78.71)	473.74 (75.54)	471 (77.52)

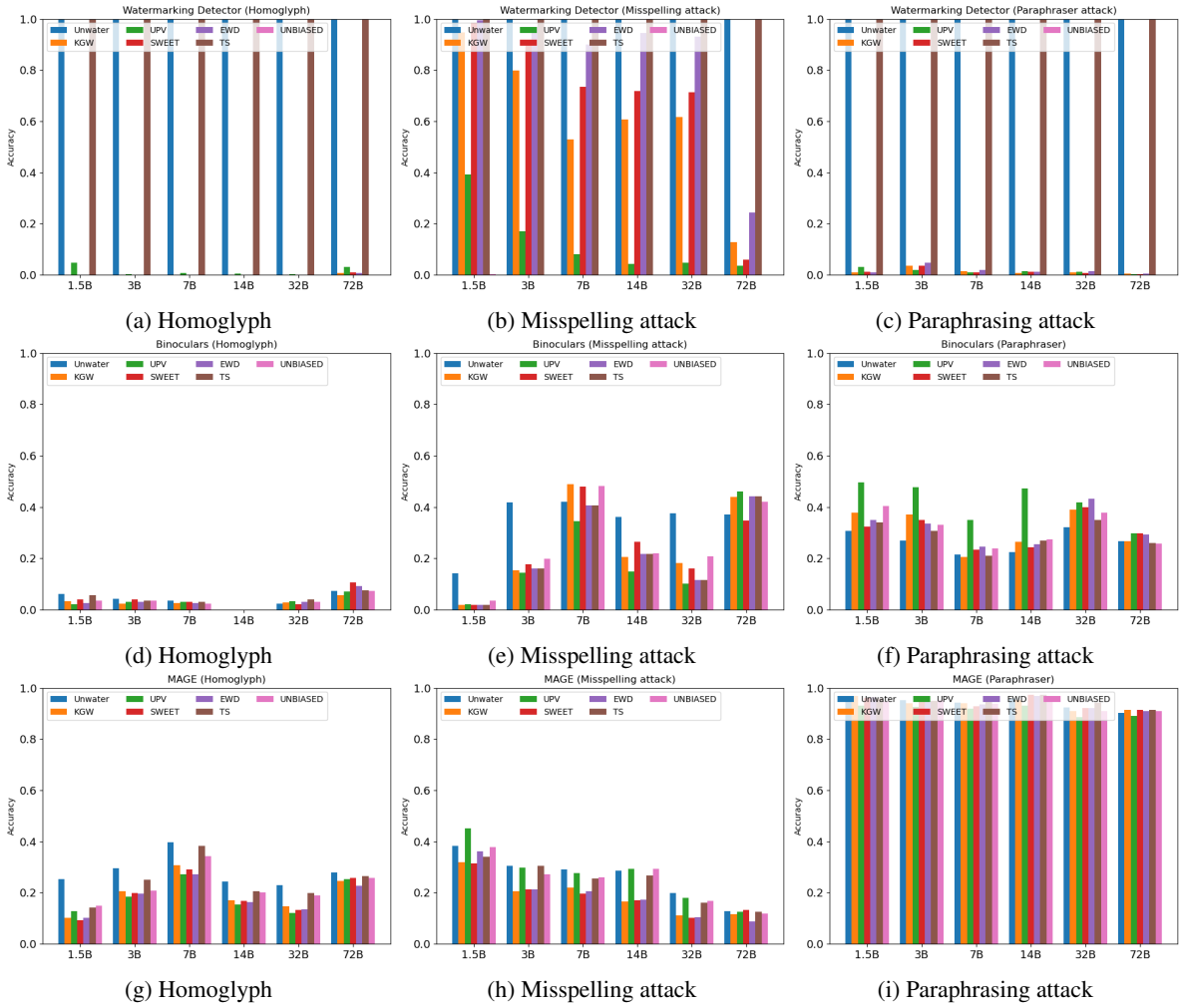
Table 6: Summary Statistics – Avg. word count (std) for **short-texts** (above) **long-texts** (below)

LLMs	Unwater	KGW	UPV	SWEET	EWD	TS	Unbiased
Qwen2.5-1.5B	**2.29** (2.933)	**2.87** (4.354)	**4.576** (5.661)	**2.526** (3.518)	**2.542** (3.566)	1.82 (2.193)	**2.2** (3.012)
Qwen2.5-3B	1.184 (0.4353)	1.180 (0.4103)	1.068 (0.2563)	1.324 (0.5558)	1.344 (0.5536)	1.171 (0.3981)	1.158 (0.3996)
Qwen2.5-7B	1.617 (1.317)	1.902 (1.092)	1.157 (0.4572)	1.773 (1.792)	1.602 (0.976)	1.722 (1.528)	1.452 (0.8856)
Qwen2.5-14B	1.046 (0.2387)	1.013 (0.1138)	1.019 (0.1667)	1.029 (0.1891)	1.032 (0.1757)	1.133 (0.4053)	**3.036** (1.6552)
Qwen2.5-32B	1.068 (0.2725)	1.115 (0.5003)	1.071 (0.3041)	1.086 (0.2808)	1.08 (0.2844)	1.114 (0.3401)	1.19 (0.4581)
Qwen2.5-72B	1.17 (0.3829)	1.232 (0.4561)	1.146 (0.3665)	1.159 (0.3737)	1.2 (0.4045)	1.205 (0.4132)	1.151 (0.3659)
LLMs	**Unwater**	**KGW**	**UPV**	**SWEET**	**EWD**	**TS**	**Unbiased**
Qwen2.5-1.5B	**33.6** (8.917)	**35.27** (6.616)	27.11 (8.733)	35.79 (6.466)	**35.87** (7.025)	**36.13** (8.255)	**32.81** (8.398)
Qwen2.5-3B	30.34 (10.05)	30.84 (8.804)	24.76 (8.576)	33.52 (12.16)	33.46 (10.81)	31.15 (11.05)	29.89 (10.8)
Qwen2.5-7B	33.18 (20.41)	32.79 (9.837)	25.72 (9.223)	34.06 (14.63)	33.39 (15.21)	31.16 (9.161)	31.85 (14.86)
Qwen2.5-14B	33.07 (10.4)	34.94 (6.619)	25.59 (6.691)	**35.88** (6.514)	33.83 (6.861)	33.83 (6.861)	32.16 (9.718)
Qwen2.5-32B	27.42 (11.53)	30.68 (7.981)	22.25 (6.028)	30.5 (10.24)	30.49 (8.517)	28 (7.462)	27.98 (10.1)
Qwen2.5-72B	31.88 (16.23)	33.4 (13.6)	**29.39** (9.439)	33.86 (22.68)	32.35 (15.95)	31.78 (12)	32.16 (15.81)

Table 7: Summary Statistics – Avg. sentence count (std) for **short-texts** (above) **long-texts** (below)

Technique	Text
KGW (clean text)	In a small town nestled among rolling hills, Alex stood nervously outside the trucking school. His heart pounded with fear as he contemplated his future. The prospect of starting anew in a different career loomed large. . . .
Homoglyph attack	In a small town nestled among rolling hills, Alex stood nervously oustide the trucking school. His heart pounded with fear as he contemplated his future. The prospect of starting anew in a different career loomed large.
Misspellings attack	In a small twon nestled amoung rolling hills, Alex stood nervously outside ther trucking school. His heart pounded with fear as he contemplated his future. Tihe prospect of starting anew in a different carreer loomed larg.
Paraphrasing attack	In a tranquil hamlet encircled by verdant hills, an individual named Alex, filled with apprehension, stood before a truck driving school. His heart thundered with anxiety as he pondered his future.

Table 8: Examples of adversarial text perturbations applied to KGW-generated text.



(a) Homoglyph (b) Misspelling attack (c) Paraphrasing attack

(d) Homoglyph (e) Misspelling attack (f) Paraphrasing attack

(g) Homoglyph (h) Misspelling attack (i) Paraphrasing attack

Figure 6: Obfuscation results for Watermark detector (first-row), Binoculars (second-row), and MAGE (third-row) detector