Statistic measures in Watermark detection

Jose Luis Martin-Navarro

April 8, 2025

Abstract

Detecting machine-generated text is a common problem due to the widespread adoption by the public of Large Language Models (LLMs). Techniques based on neural networks, information retrieval and zero-shot detection have been proposed. However, watermarking stands out due to the direct verification of the origin, and precision, compared with techniques that need a post-hoc analysis or pattern recognition to detect synthetic text.

Many watermarking methods have been proposed in the literature in the recent years, with different types of watermark embedding and detection methods. The principal metric to compare watermark methods is the accuracy in the watermark detection. In other words, which is the accuracy on discerning a text with a watermark from a text without it. However, the diversity on the watermarking approaches difficulties its comparison. Some methods use intrinsically different statistics measures for the confidence of the watermark detection, others use the same statistics measures with different assumptions on the population.

This assignment compares and analyzes the most used statistic measures in watermarking: the z-score and Area under the Receiver Operating Characteristic curve (AUROC). By understanding how these statistic measures are used on the context of watermark detection we can better compare the accuracy of different watermarking approaches.

Keywords: Watermark, LLM, z-score, AUROC

The problem of detecting synthetic text

The popularity of generative machine learning technologies such as Large Language Models (LLM) has brought a new set of challenges. One of them is synthetic text detection, or the problem of discerning whenever a text has been generated by a human (organic) or by a machine (synthetic). This is a problem with direct security implications in the form of fraud detection and LLM's collapse prevention. An example of fraud detection can be academic fraud, where a researcher submits a fake report, with experiments and results machine generated. An LLM's collapse can occur if synthetic data is used to train the model for example when using public available data.

To address this issue, a variety of synthetic text detectors have been proposed by researchers. These detectors can be grouped into four distinct approaches: neural network based detection, zero-shot detection, information retrieval and watermarking.

Neural network based detectors [13, 14] leverage deep learning models trained on huge datasets to identify patterns in synthetic text. Zero-shot detection [5, 7, 12] uses models as well, but unlike the neural network based approach, it only relies on general language understanding, not on prior training. Information retrieval detection[10] compares the data to be examined with extensive databases of known AI outputs to identify matches or similarities. Watermarking [1, 8, 4, 17], as opposed to the previous techniques, embeds identifiable markers in synthetic text, allowing a direct verification of the origin rather than a pot-hoc analysis or pattern recognition, which is influenced by the quality of available data to recognize the patterns.

The rest of this assignment focuses on watermarking for synthetic text detection. For a more complete overview on the detection technology landscape please refer to the survey by Wu et al..

Watermarking properties and processes

Its direct verification approach makes watermarking an interest technique to solve synthetic text detection, however it comes with its own set of challenges. Let's start by looking into watermarking main properties, as defined by Kirchenbauer et al.:

- Watermarks should be identified just through an algorithmic process, with no necessity of prior knowledge of the model parameters or access to the language model API. This permits open-source watermarking detection even with close-source models, with the detection being cost and time effective, as the LLM does not require loading or execution.
- The generation of the watermarks should be possible without retraining the model, making the generation time and cost effective as well.
- In addition, the detection should only use a contiguous portion of the generated text to detect the watermark. This ensures that the watermark remains detectable when only a slice of the generation is used to create a larger document.
- Related to the resilience of the watermark, the watermark should persist unless a significant fraction of the generated tokens is modified.
- Finally, the detection should be calculated with a rigorous statistical measure of confidence.

Authors have proposed watermarking techniques with different types of embedding based on these properties. An embedding technique that has gained a lot of attention comes from Kirchenbauer et al. work, which proposed the partition of the vocabulary into a "green list" and a "red list" based on a hash of the last token processed (KGW-hard) and some weights (hardness parameter) to improve the quality of the chosen tokens (KGW). This have been extended by several authors, for example by Lee et al. (SWEET), which adapts the embedding of KGW to increase the quality of the watermarked text. The hardness parameter in KGW is used in the embedding to balance the proportion of green tokens without damaging the quality of the output (for example leaving out an important token because it lands in the red list). On SWEET, tokens with low entropy are ignored instead. By carefully choosing the threshold, SWEET is able to improve the quality of the generated text, specially for code-generation models. Fairoze et al. uses a similar watermarking model that embeds first a message and then its signature, making it publicly verifiable. There are other watermarking proposals who differ from KGW, for example Golowich and Moitra include error-correcting codes as part of the watermark encoding to resist edits on the text that could remove it.

To be able to compare the different proposed watermarks, one needs to analyze their accuracy on detecting watermarked text. A popular method among researchers [8, 4, 19, 18] is to evaluate the precision of the detection by analyzing if a measurement (e.g. amount of green tokens) fits into the distribution known in organic text or not (z-score). To complete the evaluation the rate of false positives and negatives is analyzed using AUROC. Other works differ on their evaluation, for example Fairoze et al. performs the evaluation on the watermark of characters instead of tokens. Golowich and Moitra proposal has a promising theoretical result, however they do not provide an experimental evaluation. This assignment focuses on the works that use the z-score and AUROC, analyzing its usage and whenever it is possible to compare methodologies that use the same statistics on the detection.

Other evaluations have been proposed to extend the accuracy of the detection, measuring the robustness of the watermark against manipulations on the watermarked text. Kirchenbauer et al. considered the watermark precision on some input with certain level of paraphrasing done by an attacker. In Golowich and Moitra work, due to the error-correcting code using in the embedding, they calculated the edit-distance resistance of the watermark to non adversarial modifications. It is left for future work the analysis of how active manipulation on watermarked text affects the detection depending on the statistic measure.

Watermark detection statistics: z-score and Area under the Receiver Operating Characteristic curve (AUROC)

The z-score [15]is an statistics test that allows to decide between two complementary hypothesis (hypothesis and null hypothesis), and it tests the deviation from a population mean distribution that is already known. In the context of watermarking, the z-score is usually applied as a one-sample location test [8, 19, 18, 17, 11, 4]. Let's explain how it works taking KGW as an example watermark model. In KGW, from a given text sample it is possible to count the amount of tokens in the "green" list. In this case, the null hypothesis corresponds to a text that is organic. The organic text is assumed to be generated

by a person writing normally, without being aware of the green list. since the words on the list is random, it is possible to conclude that the distribution of green tokens is known, and it is roughly 50% with a big enough sample. The complementary hypothesis is that the text is watermarked. With the z-score, and given a certain error threshold, it is possible to conclude if the proportion of green tokens in a sample doesn't match the proportion of green tokens from the known organic distribution (one-sample location).

The z-score fits the problem of detecting a watermark in a given sample. However, it is important to understand how exactly can it be applied, specially if we want to compare z-scores of watermark methods that work differently. The most important assumption need to apply the z-score is that the distribution of the null hypothesis is known (organic text). In practice we should know what is the standard deviation, and measure it if there is available data.

Following is the list of some of the works that use the z-score:

- KGW[8] uses the test to compare if the distribution of the "green" tokens on the sample satisfies the null hypothesis (non-watermarked) or not (watermarked). The test can be applied here because a natural writer is expected to violate the red list rule with half of their tokens. If the null hypothesis is true, given T number of tokens, the mean of tokens in the list will be $T/2$ with a variance of $T/4$. Although a dataset of text is used to analyze the perplexity and the configuration parameter, the actual distribution of organic text is not measured.
- Zhang et al.[18] analyze the elimination of the watermark from the watermarked text through quality preserving perturbation methods. To assess their approach, they measure the drop of the the z-score in KGW [8] and Unigram [19].
- Yang et al.[17] uses a z-score with a slightly different null assumption than [8, 4]. In their case the null hypothesis is that the observed encoding occurs randomly (human generated). They measured the probability of their encoding with organic data it is close to 50%, which validates the theoretical assumption of the the null hypothesis distribution.
- SWEET [11] adapts the detection method of KGW. They provide a z-score calculation that includes the threshold as $z_{\text{threshold}}$, and predict the lower bound comparison between SWEET and KGW. However, they do not calculate if the null hypothesis mean is the same or has changed with their threshold method. This does not necessarily mean the z-score is incorrectly used, since on the AUROC evaluation their model seems to perform as predicted.
- Fu et al.[4] use the distribution of green tokens as a detection metric, like KGW and SWEET. However, their watermark embedding uses semantic knowledge to prioritize the inclusion of content related to the input into the green list partition. This explains why, by only comparing the z-score (Figure 1) their watermarking model outperforms KGW. However, their approach has an unintended consequence. By changing how the "green token" list is partitioned, from random to an informed way (with semantic knowledge), they are also affecting to the assumptions on the null hypothesis, which assume that a human generated text will have an equally probably number of tokens in and out of the list (mean $T/2$ and variance $T/4$). On their paper it is mentioned that since humans also use semantic knowledge to pick the next word they are going to write, their watermarking detection method is not necessarily better at discerning between watermarked and non-watermarked text (Figure 2). This disparity of results can be seen by comparing both Figure 1 and Figure 2. Although in Figure 1 the z-score regarding the ratio of correct tokens is higher in their proposed method Fu et al. compared with KGW, when comparing the AUROC the performance switches. Since AUROC is a more coherent aggregated classification metric, this result indicates an underlying problem with the z-score calculation.

The Receiver Operating Characteristic curve (ROC) is a plot of the true positive rate (y-axis) against the false positive rate (x-axis) at varying classification thresholds [2]. This plot illustrates the performance of a binary classification model. A classifier picking at random is depicted as the diagonal line, where results above the diagonal depict better than random classification results. Thus the closer to the upper left corner (0,1) the better the model performs.

The Area under the Receiver Operating Characteristic curve (AUROC) also seen as Area under the curve (AUC) represents the probability that the model, given a randomly chosen positive and negative example, will rank the positive higher than the negative. It is also called the degree of separability.

It is a measurement used in the machine learning com-

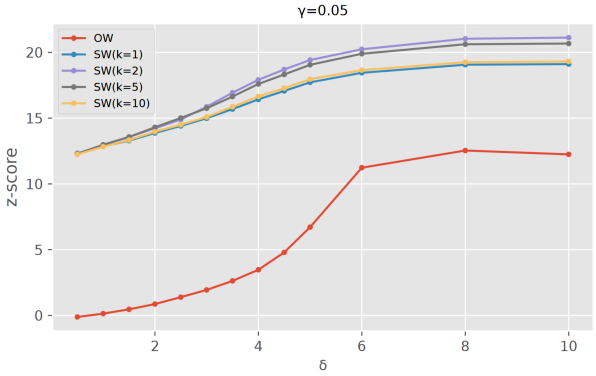

Figure 1: Watermark detection: Average z-score under different δ settings (x-axis). The hardness parameter δ affects how much the watermarking is allowed to deviate from the green token list. A high δ makes the watermark easier to detect but decreases the quality of the output. The red dataset (OW) is [8] model meanwhile the others (SW) correspond to [4]. Image from [4]

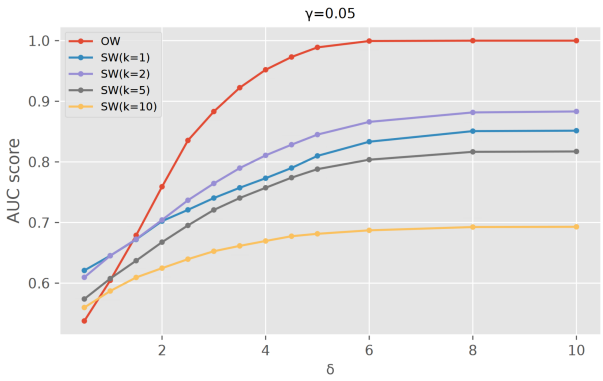

Figure 2: Watermark detection: AUC scores under different δ settings. Higher AUC scores indicates a better detection performance. Image from [4]

munity for model comparison due to the coherence on the aggregated classification performance. Since the detection of the watermarked can be phrased as a binary classification problem, the AUROC is used in watermark papers [4, 9, 11] to represent the sensitivity of the resulting hypothesis, complementing the detection statistics of the z-score.

However, AUROC is not a perfect indicator. It can be noisy, and leaves out the negative predictive value, since it only compares the true positive rate against the false positive rate [2].

In the case of Fu et al., using the AUROC allows the detection of a mismatch with the performance of the z-score metric, but it is not enough to identify the underlying issue.

Conclusions

This assignment presents the analysis of different watermarking methods that use the z-score statistic measurement for watermarking detection. By analyzing the specifics of the z-score, its assumptions, and how it is used by watermark detectors, we gained a better understanding on its usage for watermarking and when it is fair to use it to compare models [8, 4, 19].Although most of the works use the z-score correctly, we identified an incorrect usage of the z-score in Fu et al., where the informed partition of the token list affected also the null hypothesis the z-score is based on. Although the disparity on the watermarking detection precision is acknowledged in the paper, they attribute it to the fact that humans also use semantic knowledge to decide which word they write after another.

It is left for future work the measurement of the null hypothesis mean and variance values for the semantic partition proposed by Fu et al.. By checking how the semantic partition affect the mean and variance, a better metric can be used to compare their work with Kirchenbauer et al. and others using the z-score.

References

[1] Scott Aaronson and Hendrik Kirchner. Watermarking gpt outputs, 2022. URL `https://www.scottaaronson.com/talks/watermark.ppt`.

[2] Davide Chicco and Giuseppe Jurman. The Matthews correlation coefficient (MCC) should replace the ROC AUC as the standard metric for assessing binary classification. *BioData Mining*, 16(1):4, February 2023. ISSN 1756-0381. doi: 10.1186/s13040-023-00322-4. URL `https://doi.org/10.1186/s13040-023-00322-4`.

[3] Jaiden Fairoze, Sanjam Garg, Somesh Jha, Saeed

Mahloujifar, Mohammad Mahmoody, and Mingyuan Wang. Publicly-Detectable Watermarking for Language Models, May 2024. URL http://arxiv.org/abs/2310.18491. arXiv:2310.18491 [cs].

[4] Yu Fu, Deyi Xiong, and Yue Dong. Watermarking Conditional Text Generation for AI Detection: Unveiling Challenges and a Semantic-Aware Watermark Remedy. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):18003–18011, March 2024. ISSN 2374-3468. doi: 10.1609/aaai.v38i16.29756. URL https://ojs.aaai.org/index.php/AAAI/article/view/29756. Number: 16.

[5] Sebastian Gehrmann, Hendrik Strobelt, and Alexander M. Rush. GLTR: Statistical Detection and Visualization of Generated Text, June 2019. URL http://arxiv.org/abs/1906.04043. arXiv:1906.04043 [cs].

[6] Noah Golowich and Ankur Moitra. Edit Distance Robust Watermarks for Language Models, 2024.

[7] Daphne Ippolito, Daniel Duckworth, Chris Callison-Burch, and Douglas Eck. Automatic Detection of Generated Text is Easiest when Humans are Fooled, May 2020. URL http://arxiv.org/abs/1911.00650. arXiv:1911.00650 [cs].

[8] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. A Watermark for Large Language Models. In *Proceedings of the 40th International Conference on Machine Learning*, pages 17061–17084. PMLR, July 2023. URL https://proceedings.mlr.press/v202/kirchenbauer23a.html. ISSN: 2640-3498.

[9] John Kirchenbauer, Jonas Geiping, Yuxin Wen, Manli Shu, Khalid Saifullah, Kezhi Kong, Kasun Fernando, Aniruddha Saha, Micah Goldblum, and Tom Goldstein. On the Reliability of Watermarks for Large Language Models, May 2024. URL http://arxiv.org/abs/2306.04634. arXiv:2306.04634 [cs].

[10] Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. Paraphrasing evades detectors of AI-generated text, but retrieval is an effective defense, October 2023. URL http://arxiv.org/abs/2303.13408. arXiv:2303.13408 [cs].

[11] Taehyun Lee, Seokhee Hong, Jaewoo Ahn, Ilgee Hong, Hwaran Lee, Sangdoo Yun, Jamin Shin, and Gunhee Kim. Who Wrote this Code? Watermarking for Code Generation, July 2024. URL http://arxiv.org/abs/2305.15060. arXiv:2305.15060 [cs].

[12] Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. DetectGPT: Zero-Shot Machine-Generated Text Detection using Probability Curvature, July 2023. URL http://arxiv.org/abs/2301.11305. arXiv:2301.11305 [cs].

[13] OpenAI. New AI classifier for indicating AI-written text, 2023. URL https://openai.com/index/new-ai-classifier-for-indicating-ai-written-text/.

[14] Muhammad A. Panhwar, Kamran A. Memon, Adeel Abro, Deng Zhongliang, Sijjad A. Khuhro, and Saleemullah Memon. Signboard Detection and Text Recognition Using Artificial Neural Networks. In *2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, pages 16–19, July 2019. doi: 10.1109/ICEIEC.2019.8784625. URL https://ieeexplore.ieee.org/document/8784625/?arnumber=8784625. ISSN: 2377-844X.

[15] Maria Dolores Ugarte, Ana F. Militino, and Alan T. Arnholt. *Probability and Statistics with R*. CRC Press, April 2008. ISBN 978-1-58488-892-5. Google-Books-ID: 0zjcBQAAQBAJ.

[16] Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Derek F. Wong, and Lidia S. Chao. A Survey on LLM-Generated Text Detection: Necessity, Methods, and Future Directions, April 2024. URL http://arxiv.org/abs/2310.14724. arXiv:2310.14724 [cs].

[17] Xi Yang, Kejiang Chen, Weiming Zhang, Chang Liu, Yuang Qi, Jie Zhang, Han Fang, and Nenghai Yu. Watermarking Text Generated by Black-Box Language Models, May 2023. URL http://arxiv.org/abs/2305.08883. arXiv:2305.08883 [cs].

[18] Hanlin Zhang, Benjamin L. Edelman, Danilo Francati, Daniele Venturi, Giuseppe Ateniese, and Boaz Barak. Watermarks in the Sand: Impossibility of Strong Watermarking for Generative Models, November 2023. URL http://arxiv.org/abs/2311.04378. arXiv:2311.04378 [cs].

[19] Xuandong Zhao, Prabhanjan Ananth, Lei Li, and Yu-Xiang Wang. Provable Robust Watermarking for AI-Generated Text, October 2023. URL http://arxiv.org/abs/2306.17439. arXiv:2306.17439 [cs].