UNDERSTANDING THE IMPACT OF WATERMARK REMOVAL PROMPTS ON LLM PERFORMANCE

Tairan Fu
College of Mechanical and Electrical Engineering
Nanjing University of Aeronautics and Astronautics
Nanjing, China
ftr258@nuaa.edu.cn

Javier Conde and Pedro Reviriego
ETSI de Telecomunicación
Universidad Politécnica de Madrid
Madrid, Spain
{javier.conde.diaz, pedro.reviriego}@upm.es

ABSTRACT

As Artificial Intelligence-generated content becomes common, there is a growing interest in identifying text generated by Large Language Models (LLMs). To do this, major LLM providers are designing and introducing waterwarking techniques in their models. At the same time, schemes are being proposed to remove those watermarks, for example, by using prompts that force the LLM to generate text that contains additional characters that are subsequently eliminated to remove the watermark. These prompting techniques have been shown to be effective in removing some watermarks, but since they significantly modify the text generation process, they can affect the performance of the LLM. In this work, the impact of watermark removal prompts on the performance of state-of-the-art proprietary models is evaluated. Two different evaluation benchmarks are run with the original and attack prompts. The results show that attack prompts do not have a significant impact on the results, which are in some cases even better when using an attack prompt than the original one. This suggests that advanced LLMs are not affected by attack prompts, and thus removal of watermarks will have a limited impact on the quality of the generated text.

***K*eywords** LLMs · Watermark · Attack · Performance

1 Introduction

The massive adoption of Large Language Models (LLMs) is making LLM-generated text common not only on the Internet, but also in almost any type of document, such as student assignments, project reports, and administrative documentation [1]. This creates many challenges; for example, it is difficult to be sure that an assignment, a paper, a blog post, or a news has been written by the author and not by an LLM [2, 3, 4]; the text generated by LLMs is likely to be scraped and used to train newer LLMs creating a loop [5] that can only be avoided if LLM-generated content can be detected; LLM providers have no way to prove that text has been generated by their models and the list of issues continues. The bottom line is that mechanisms are needed to detect LLM-generated text [6].

Several techniques have been proposed to detect LLM-generated content [1], which can be divided into two main groups: those that rely only on LLM-generated text without having access to modify how the models generate the text and those that modify the generation process [4]. The first group considers the LLM as a black box, typically builds a dataset of LLM-generated text and human text and tries to find differences, for example, in linguistic patterns or in the frequency of use of words and expressions. Then those features are used as input to a classifier that, given a text, categorizes it as human-generated or LLM-generated [7]. These black box techniques rely on the features of the text generated by the LLM and thus have to be revised and adjusted each time a new model or a new version of an LLM is released, and also face the problem of the variability in the text generated by an LLM when different parameters such as temperature or system prompt are used [8]. Therefore, black-box schemes are not scalable and tend to be error prone. An alternative is to modify the LLM generated text to include a watermark that can be used later for detection [9]. This can be done by first generating the text with the LLM and then manipulating it to introduce the watermark or by changing the way the LLM generates the text so that a watermark is included in all the text it generates [6].

The insertion of a watermark as part of text generation is a scalable and model-independent solution that has been implemented by large companies [10]. The basic mechanism to insert the watermark is to modify the sampling of the next token, introducing patterns that can be detected [11]. For example, for each new token, we may restrict the choices to one-half of the tokens in the dictionary and make the selection of the allowed tokens depend on the previous tokens using a hash function so that it is different for each token. Then the text generated by such a procedure can be detected by computing the hash, applying the selection, and checking if the tokens in the text are allowed. This strategy can be refined by introducing more sophisticated and subtle changes in the next token sampling process that do not introduce noticeable patterns or changes in the generated text [12]. In fact, this has been proved both theoretically and by large-scale evaluations with humans [10]. Therefore, the use of watermark insertion at inference time when text is generated is attractive to service providers.

As with traditional watermarks, an important feature is how difficult it is to remove the watermark from the text so that it is not detected [13]. For watermarks that work by modifying the sampling for a token based on the previous ones (a hash function of these controls the changes in sampling), several attacks have been proposed to remove the watermark. For example, the generated text can be rephrased so that if most tokens are changed, the watermark cannot be detected anymore [14]. However, rephrasing the text requires the use of another LLM which may have a lower quality or another watermark. Other attacks modify the generated text, for example, by using contractions in verbs or changing words by synonyms or by introducing misspellings or typos [14]. Finally, the LLM itself can be exploited to remove the watermark, for example, by prompting the model to insert emojis between words [14] when generating the text. Then those emojis are removed to extract the text, and in doing so the watermark is removed too. The last method is simple and can be easily implemented with different prompts.

Attacks on watermarks have an impact on the final text and may degrade its quality. For attacks that manipulate the generated text, adding typos or verb contractions, the choices of the attacker are rather limited, and the impact on text is easy to understand. Instead, for prompt-based attacks, we can ask the LLM to do many different things, for example, add additional spaces instead of emojis or answer in capital letters. Additionally, the impact of each of those options on the quality of the text generated by the LLM is not intuitive and will depend on the model. Therefore, it is of interest to analyze the impact on text quality of different prompt strategies to remove watermarks.

In this paper, the impact of different prompts to remove watermarks on the generated text is evaluated on state-of-the-art proprietary LLMs. The main findings of the study are that watermark removal prompts have limited impact on LLM performance and in some cases the results are even better than with the original prompt.

The rest of the paper is organized as follows, Section 2 presents the methodology used discussing the attacks, benchmark, and LLMs considered in the evaluation. The results are presented in section 3 and discussed in section 4. The paper ends with the conclusion in Section 5.

2 Methodology

This section discusses the attack strategies considered, the LLMs evaluated, and the benchmarks used in our study.

2.1 Attacks

The attacks prompts considered in the literature focus on the insertion of emojies between words [14]. In our study, we also consider alternative prompting strategies. The attacks evaluated are the following.

1. Emoji insertion: models are asked to insert an emoji between words [14].
2. Space insertion: models are asked to insert an extra space between words.
3. Capital letters: models are asked to respond only in capital letters.

The rationale for the first two attacks is to insert characters between words that do not alter the meaning and may have been present in the training dataset. The last attack tries to exploit the fact that capital letters may also have been present during training and do not change the meaning of the text. Intuitively, the performance of the LLM would be less impacted if the attack introduces modifications that are less noticeable in the generated text.

2.2 LLMs

The prompts force the model to produce texts that deviate from a normal output and thus pose an additional challenge. Intuitively, models with more capability will suffer a lower degradation than models with lower performance. At the same time, there is more incentive to attack high-performance proprietary models as these cannot be run locally and produce high-quality text. For models that are run locally and for which the code is open, removing the watermark generation part is trivial. Therefore, in our evaluation, we focus on proprietary models of different sizes and with different performance to assess how those factors impact the ability of the models to generate quality text with the attack prompts. In more detail, the following models are evaluated:

1. GPT4o and GPT4o-mini from OpenAI [15].

2. Claude 3.5 Sonnet and Claude 3 Haiku from Anthropic.

2.3 Benchmark and procedure

To evaluate the impact of attack prompts on the performance of LLMs, we use the MMLU-Pro benchmark [16] with chain-of-thought prompts. This benchmark is an evolution of the original Massive Multitask Language Understanding (MMLU) [17] benchmark and is intended to measure the ability of LLM to understand and answer multiple choice questions on a wide range of topics. The questions have been carefully selected and reviewed, and the performance of the models is significantly better when using a chain of thought compared to a direct answer [16]. Therefore, to answer correctly, the model requires in many cases a reasoning text. This is suitable for our evaluation as a performance degradation in the generated text due to the attack prompt will most likely affect the accuracy of LLM responses. In addition, the Graduate-Level Google-Proof Q&A (GPQA) Benchmark [18] is also run to check if the attack prompts impact the accuracy on difficult science questions.

The evaluation procedure is thus as follows: MMLU-Pro and GPQA are run for the original, and attack prompts instruct the model to reason before answering, and the accuracy of the models is measured. The metric used to quantify the impact of the attack prompts is the loss of accuracy compared to the original prompt.

3 Evaluation

The results of MMLU-pro are summarized in Figure 1, which shows the precision of the models for each of the prompts on all MMLU-pro questions. It can be seen that the accuracy is similar for the original and three attack prompts. The results for GPQA are shown in Figure 2 and again the differences are small and, in some cases, the accuracy is even better for the attack prompts. These results suggest that watermark removal prompts have little impact on advanced state of the art LLM performance.

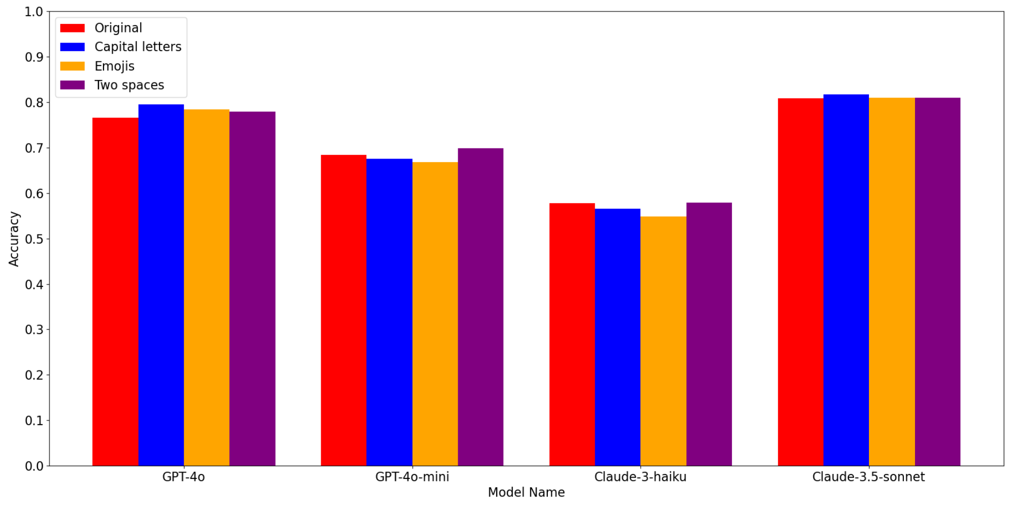

Figure 1: Accuracy on the MMLU-Pro for the different models and attack prompts.

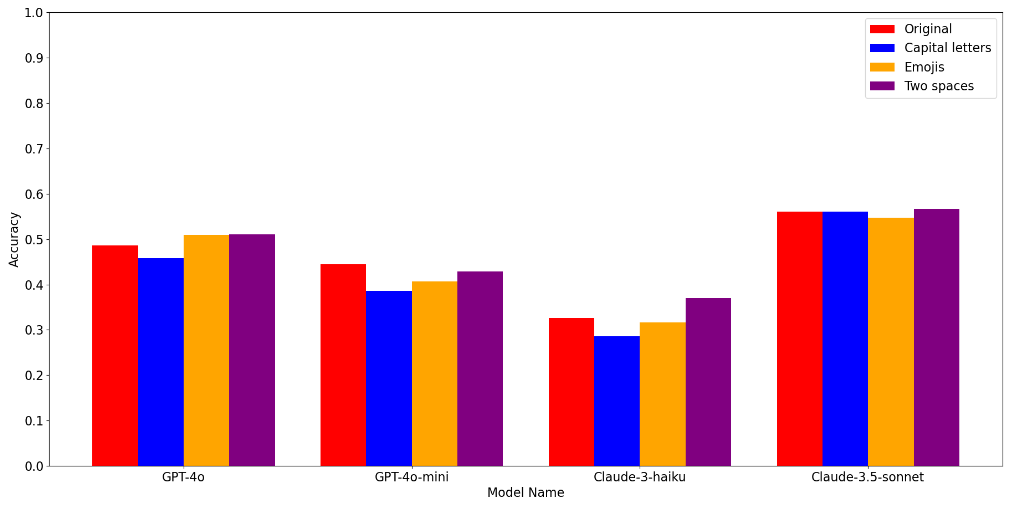

Figure 2: Accuracy on the GPQA for the different models and attack prompts.

4 Limitations

The evaluation presented in the previous section has a number of limitations. The first is that performance has only been evaluated in English, but the majority of LLM users are not native English speakers [19]. Another limitation is that the quality of the generated text is measured indirectly by its impact on the accuracy of the responses to multiple choice questions. Therefore, ideally, the evaluation should be extended to other languages, and human ratings should be included to assess the quality of the text. Finally, as with any evaluation, additional attack prompts and LLMs could be tested.

5 Conclusion

This paper has evaluated the impact of different prompts designed to remove watermarks on the quality of the text generated by LLMs. The results suggest that the performance of advanced state of the art proprietary LLMs is not affected by watermark removal prompts.

References

[1] Junchao Wu, Shu Yang, Runzhe Zhan, Yulin Yuan, Lidia Sam Chao, and Derek Fai Wong. A survey on llm-generated text detection: Necessity, methods, and future directions. *Computational Linguistics*, pages 1–65, 2025.

[2] Michael Sheinman Orenstrakh, Oscar Karnalim, Carlos Anibal Suarez, and Michael Liut. Detecting llm-generated text in computing education: Comparative study for chatgpt cases. In *2024 IEEE 48th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 121–126. IEEE, 2024.

[3] Canyu Chen and Kai Shu. Can llm-generated misinformation be detected? In *NeurIPS 2023 Workshop on Regulatable ML.*

[4] Javier Conde, Pedro Reviriego, Joaquín Salvachúa, Gonzalo Martínez, José Alberto Hernández, and Fabrizio Lombardi. Understanding the impact of artificial intelligence in academic writing: Metadata to the rescue. *Computer*, 57(1):105–109, 2024.

[5] Gonzalo Martínez, Lauren Watson, Pedro Reviriego, José Alberto Hernández, Marc Juarez, and Rik Sarkar. Towards understanding the interplay of generative artificial intelligence and the internet. In *International Workshop on Epistemic Uncertainty in Artificial Intelligence*, pages 59–73. Springer, 2023.

[6] Ruixiang Tang, Yu-Neng Chuang, and Xia Hu. The science of detecting llm-generated text. *Commun. ACM*, 67(4):50–59, March 2024.

[7] Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. GLTR: Statistical detection and visualization of generated text. In Marta R. Costa-jussà and Enrique Alfonseca, editors, *Proceedings of the 57th Annual Meeting*

of the Association for Computational Linguistics: System Demonstrations, pages 111–116, Florence, Italy, July 2019. Association for Computational Linguistics.

[8] Gonzalo Martínez, José Alberto Hernández, Javier Conde, Pedro Reviriego, and Elena Merino-Gómez. Beware of words: Evaluating the lexical diversity of conversational llms using chatgpt as case study. *ACM Trans. Intell. Syst. Technol.*, September 2024. Just Accepted.

[9] Sahar Abdelnabi and Mario Fritz. Adversarial watermarking transformer: Towards tracing text provenance with data hiding. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 121–140, 2021.

[10] Sumanth Dathathri, Abigail See, Sumedh Ghaisas, Po-Sen Huang, Rob McAdam, Johannes Welbl, Vandana Bachani, Alex Kaskasoli, Robert Stanforth, Tatiana Matejovicova, et al. Scalable watermarking for identifying large language model outputs. *Nature*, 634(8035):818–823, 2024.

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

[12] Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. Robust distortion-free watermarks for language models. *arXiv preprint arXiv:2307.15593*, 2023.

[13] 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. *arXiv preprint arXiv:2311.04378*, 2023.

[14] Zesen Liu, Tianshuo Cong, Xinlei He, and Qi Li. On evaluating the performance of watermarked machine-generated texts under adversarial attacks. *arXiv preprint arXiv:2407.04794*, 2024.

[15] OpenAI. Gpt-4 technical report, 2023.

[16] Yubo Wang, Xueguang Ma, Ge Zhang, Yuansheng Ni, Abhranil Chandra, Shiguang Guo, Weiming Ren, Aaran Arulraj, Xuan He, Ziyan Jiang, et al. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. *arXiv preprint arXiv:2406.01574*, 2024.

[17] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[18] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R Bowman. Gpqa: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2023.

[19] Klaudia Thellmann, Bernhard Stadler, Michael Fromm, Jasper Schulze Buschhoff, Alex Jude, Fabio Barth, Johannes Leveling, Nicolas Flores-Herr, Joachim Köhler, René Jäkel, et al. Towards multilingual llm evaluation for european languages. *arXiv preprint arXiv:2410.08928*, 2024.