Expanding on LLM watermarking

Bradley Liu

Table of Contents

Table of Contents

1. Introduction

As the development of large language models (LLMs) becomes more sophisticated, it becomes more difficult to differentiate between synthetic data and human-generated text. The capacity of these models to emulate human capabilities raises concerns across various domains, including social media manipulation, dissemination of misinformation, and academic integrity in writing and coding tasks (Bergman et al., 2022; Mirsky et al., 2023). As a result, the ability to identify machine generated text is of great importance in mitigating any potential harm associated with large language models.

1.1 Language Model Basics

We focus on language models for next word prediction, where we define the model as a function f which takes in an input sequence of known tokens (s^{-Np} , · · · , s^{t-1}). This sequence contains the prompt and the first t-1 tokens already produced by the language model. The function f will output a vector of logits l(t) which, using the softmax operator, gives a probability vector p(t) over the vocabulary for sampling the next token at position t. We use greedy sampling by selecting the token with the overall highest converted probability for the rest of the paper.

2. Background Knowledge

We will cover the existing watermarking schemes covered in A Watermark for Large Language Models (Kirchenbauer, 2023). Later in this paper, we will expand on the existing approach to produce an alternative watermarking scheme.

2.1 Current Watermarking Schemes

Hard Watermark. We define the simplest current watermarking scheme as the following:

Apply the language model to prior tokens (s^{-Np}… s^{t-1}) to get a probability vector pt over the vocabulary

Compute a hash of the last token s^{t-1} and use it to seed a random number generator

Using this seed, randomly partition the vocabulary into a “green list” G and “red list” R of equal size

Sample s^t from G, never generating any token in the red list

We call this a “hard” watermark, as only green list tokens are strictly selected. As a result, no red list tokens will be found in the watermarked, and instead all output tokens will belong to the green list. Though this scheme offers a simplistic approach to watermarking, it has drawbacks in text quality as higher quality tokens that happen to belong to the red list are never selected.

Soft Watermark.A more sophisticated approach proposed in the paper: A Watermark for Large Language Models [1] builds on the “hard” watermark algorithm explained in the previous section. Like the “hard” watermark, we categorize tokens sampled from a red and green list that together comprise the entire LLM vocabulary. Similarly a logits vector is converted into a probability vector by using the softmax operator and a word is then sampled from this vector. Unlike the hard watermark, this new “soft” watermark algorithm will promote the selection of green list tokens by a parameter ▯ > 0 instead of strictly selecting green list tokens. As a result, the watermarked text is able to contain both green and red list tokens. Due to the adjustable bias parameter ▯ , we’ll see variable higher occurrences of green list tokens in the watermarked text. The algorithm for this current algorithm given input text (s^{-Np}… s^{t-1}), green list size ɣ ∈ (0, 1) and hardness parameter ▯ > 0 can be described as follows:

Apply the language model to prior tokens (s^{-Np}… s^{t-1}) to get a logit vector lt over the vocabulary

Compute a hash of the last token s^{t-1} and use it to seed a random number generator

Using this seed, randomly partition the vocabulary into a “green list” G of size ɣ|V| and a “red list” R of size (1 - ɣ)|V|

Add ▯ to each green list logit. Apply the softmax operator to these modified logits to get a probability distribution over the vocabulary

Sample the next token, st using the watermarked distribution pt

The “soft” watermark tackles the downsides of text quality in the “hard” watermark. Higher quality tokens that belong to the red list are given a probability of being sampled compared to a strict complete avoidance of the same higher quality red list tokens.

2.2 Current Watermarking Detection

Keeping in mind that the vocabulary is partitioned randomly at each token generation, we expect human written text to violate the red list rule for the “hard” watermark. Therefore, given that a human writer produces T tokens, the probability of not violating the red list rule is $(½)^T$. The following null hypothesis is tested to detect both a “hard” and “soft” watermark:

H_0 = The text sequence is human generated

A one proportion z-test is chosen to evaluate the null hypothesis. For a true null hypothesis, the number of green list tokens detonated |s|G has expected value T/2 and variance T/4. We reject the null hypothesis if the z-score is above a chosen threshold.

The z-statistic is then the following:

$$z = 2(|s|_G - T/2)/\sqrt{T}.$$

The detection process for “soft” watermark is identical to the “hard” watermark, with the inclusion of a arbitrary green list size parameter γ

$$z = (|s|_G - \gamma T)/\sqrt{T\gamma(1-\gamma)}.$$

3. Red and Green List Soft Watermarking

We expand on the original soft watermarking algorithm proposed in the paper: A Watermark for Large Language Models (Kirchenbauer, 2023). The original soft watermark in the paper explores promoting sampling of greenlist tokens only. We expanded on an alternate approach that promotes both the sampling of green list and red list tokens, depending on which list the previous token sampled belongs to. This approach leads to two scenarios of token sampling, one in which we select a token with a different bias depending on if the last token was from the red list or the last token was from the green list respectively.

3.1 Algorithm and Proof of Concept

Just like the original implementation, the prompt is fed as input to the algorithm. The last token of the input is taken to seed a random number generator, which will be used to randomly partition the vocabulary list. Depending on a parameter gamma that specifies the fraction of green list tokens in the vocabulary, the green list tokens and the red list tokens are split at an index of the vocabulary list that satisfies this parameter gamma. Similarly as well, the last layer of the language model outputs a vector of logits l(t) which, using the softmax operator, gives a probability vector p(t) over the vocabulary for sampling output tokens.

$$p_k^{(t)} = \exp(l_k^{(t)}) / \sum_i \exp(l_i^{(t)}).$$

The difference in our proposed algorithm is the incorporation of both a greenlist bias and an added red list bias to the logits vector. This in turn will bias sampling either green/red list tokens from the probability vector generated by the specified softmax operator. For this implementation, we will retain the color of the current token with every token generation of the output. If the current token is green, we give a bias ⍰ to sampling the next token from the green list. If the current token is red, we add a bias alpha to sampling the next token from the red list.

Red and Green List and Soft Watermarking Algorithm Pseudocode

```
Input: prompt, $s^{(-N_p)} \cdots s^{(-1)}$
       green list size, $\gamma \in (0,1)$
       hardness parameter, $\delta > 0$
for $t = 0, 1, \cdots$ do
  1.  Apply the language model to prior tokens
      $s^{(-N_p)} \cdots s^{(t-1)}$ to get a logit vector $l^{(t)}$ over
      the vocabulary.
  2.  Compute a hash of token $s^{(t-1)}$, and use it to
      seed a random number generator.
  3.  Using this random number generator, randomly
      partition the vocabulary into a "green list" $G$ of
      size $\gamma|V|$, and a "red list" $R$ of size $(1-\gamma)|V|$.
```

4. Store the current green token mask over the vocabulary as a means of recognizing which list the next token s^t generated belongs to

5. Add δ to each green list logit if s^{t-1} is a green list token. Add alpha to each red list logit if s^{t-1} is a red list token. Apply the softmax operator to these modified logits to get a probability distribution over the vocabulary

$$\hat{p}_k^{(t)} = \begin{cases} \frac{\exp(l_k^{(t)}+\delta)}{\sum_{i \in R} \exp(l_i^{(t)})+\sum_{i \in G} \exp(l_i^{(t)}+\delta)}, & k \in G \\ \frac{\exp(l_k^{(t)})}{\sum_{i \in R} \exp(l_i^{(t)})+\sum_{i \in G} \exp(l_i^{(t)}+\delta)}, & k \in R. \end{cases}$$

6. Sample the next token, s^t, using the watermarked distribution $p^{(t)}$.

end for

3.2 Detection of red and green soft watermarking

We propose a simple method to detect this new watermarking scheme. We notice 4 patterns emerge from this watermark, being the sequences red green, green red, green green, and red red. This new watermarking scheme will promote the occurrences of red red and green green sequences, whereas a human written text will have an evenly distributed occurance of all four sequences. We count the number of occurrences in the LLM output of the 4 colour pattern bigram sequences. The bigram sequence occurrences are normalized over the total count of all 4 colour pattern bigram sequences and used to determine whether or not the output text is watermarked. The detection of the watermark is determined by testing the following null hypothesis,

H0: The text sequence is generated with no knowledge of the green and red list rule

We can use a one proportion z-test for a more robust detection approach, similarly to the existing soft watermarking detection process. We define $|s|_{GG,RR}$ as the total number of consecutive bigram color sequences in the output text, and T as the total number of tokens in the output text. A human source has a probability of (0.5)*(0.5) + (0.5)*(0.5) in producing consecutive green and red tokens. Fix a green list size parameter γ of 0.5, we arrive at the following z-statistic identical to the existing hard watermarking scheme.

$z = (|s|_{GG,RR} - (\gamma^2 + (1-\gamma)^2)T)/ \text{sqrt}(T(\gamma^2 + (1-\gamma)^2)(\gamma^2 + (1-\gamma)^2))$
$= (|s|_{GG,RR} - T * 0.5) / \text{sqrt}(T * 0.25)$
$=2(|s|_{GG,RR} - T/2) / \text{sqrt}(T)$

If the z-score is above a set threshold, we'll reject the null hypothesis and classify the output text as watermark text. Consequently, if the z-score is below a threshold we accept the null hypothesis and classify the text as human generated. Note as this algorithm is expanded from the current soft watermarking scheme, its detection quality is also dependent on the entropy of the text. An alternate testing strategy is given but not experimented with due to time constraints.*
*Alternate testing strategy for other values of green list size parameter γ expanded on in Limitations and Future Work

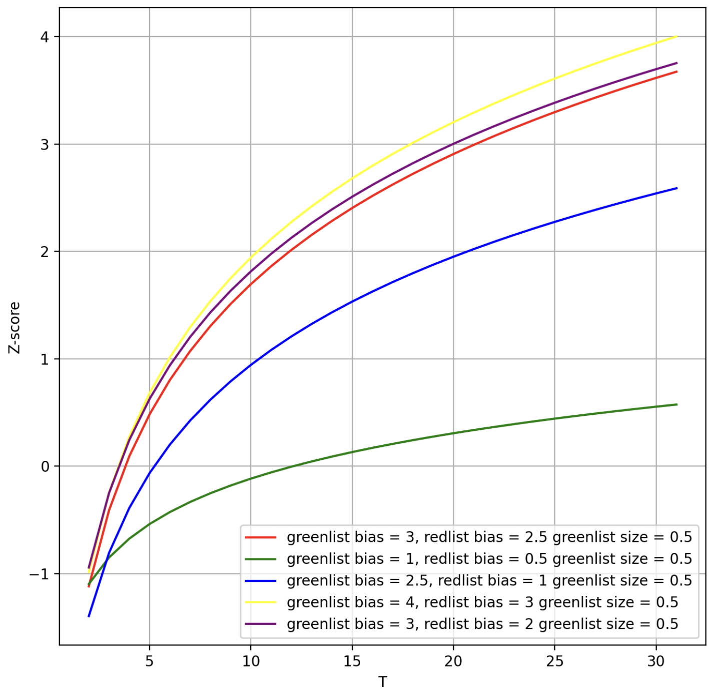
4. Experiments and Results

We analyze the new proposed watermarking scheme using facebook/opt-1.3b language model and the prompts for testing are sliced and diced from a subset of Hugging Face's C4 realnewslike dataset.

4.1 Trade off between number of tokens and watermark strength

With single prompt iterations, we graph the average z score obtained from the previous detection method stated, as a function of the token length T of watermarked generated text. For every token output length T=(1, …,30), 10 prompts are randomly sampled from the C4 dataset to calculate the average z score.

Figure 1 Effect on z-score by red list and green list bias parameters, under multinomial sampling



We see that text containing a small number of tokens will have a lower and sometimes negative z-score for our experimental adjustment of the bias.* This can be eliminated by significantly increasing the bias for both green and red list tokens but we keep in mind the tradeoff in generated text quality. We see that the experiments support the theory that the type 1 and type 2 error rates decay to 0 for increasing sequences of T.

4.2 Performance and Sensitivity for Multinomial Sampling

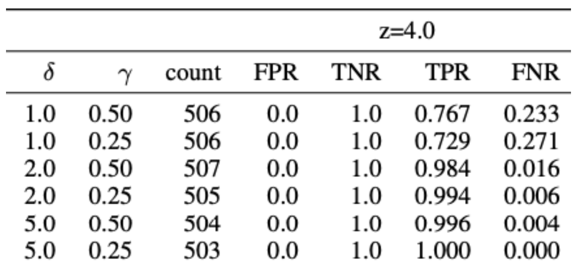
Table 1 Empirical error rates for expanded soft watermarking and detection using multinomial sampling for 50 samples

Green list bias	Red list bias	Gamma	Z-score	FPR	TNR	TPR	FNR
2	1	0.5	3	0.0	1.0	0.462	0.538
2.5	2	0.5	3	0.0	1.0	0.72	0.263
3	2	0.5	3	0.0	1.0	0.789	0.211
5	3	0.5	3	0.0	1.0	0.889	0.111
9	7	0.5	3	0.0	1.0	1.0	0.0

Error rate for different watermarking parameters is shown in Table 1 showing the sensitivity of the hypothesis test. Increasing values of green list bias and red list bias result in smaller detection errors.

5. Findings and Discussion

Table 2 Empirical error rates for existing soft watermarking and detection



			z=4.0			
δ	γ	count	FPR	TNR	TPR	FNR
1.0	0.50	506	0.0	1.0	0.767	0.233
1.0	0.25	506	0.0	1.0	0.729	0.271
2.0	0.50	507	0.0	1.0	0.984	0.016
2.0	0.25	505	0.0	1.0	0.994	0.006
5.0	0.50	504	0.0	1.0	0.996	0.004
5.0	0.25	503	0.0	1.0	1.000	0.000

We find that the expanded soft watermark performs well, but unfortunately still performs behind the existing soft watermark. Table 2 contains the error rates for the existing watermark as comparison. Identical to the existing soft watermark, we achieve a false positive rate of 0.0 and a true negative rate of

1.0. Unfortunately we need higher red and green list biases in order to achieve a perfect detection rate which leads to more loss in generated text quality. We see that small red and green list biases still give a considerable amount of correct detections, but the existing soft watermark has higher true positive rates and lower false negative rates. Note the underperformance is partially due to not having enough samples for the error rates converge to better values but after experimentation with an increased number of samples with an fixed watermark parameter, the expanded algorithm still underperforms the existing algorithm.

6. Conclusion

The expanded soft watermark remains practical and simple to detect as no access is needed to the underlying mode and it retains the text quality of the existing soft watermark. This expansion on soft watermarking showcases another potential solution with close performance to the existing soft watermark. Though the expanded soft watermark performs slightly behind the existing soft watermark, there's potential for improvement by formulating a better hypothesis test. Furthermore, due to evaluating four different bigram sequences, the expanded soft watermarking scheme offers more insight and relationships between sequences compared to the original 2 different token colours. Hopefully, these results spark interest and curiosity into the expanded soft watermarking scheme, and further optimizations and changes could be made on top to create more watermarks for fighting against unethical and malicious use of large language models.

7. Limitations and Future Work

Due to large time consumption of the language model text generation, our experiments are limited to smaller samples under time constraint. We retrieve a minimum of 30 samples to satisfy utilizing a z-test instead of a t-test but more sampling is needed for more concise and accurate findings. Furthermore perplexity was not taken into consideration during experimentation, but it is important to expand on this implementation and evaluate the correlation between the z–score and perplexity.

Alternate hypothesis testing. Instead of considering the occurrences of total green green and red red bigram sequences, it might be better to develop statistical tests considering the individual occurrences of every 4 bigram colour sequences. The green and red list biases are set to be unequal to each other in a way to promote a higher occurrence of either green or red consecutive bigrams. Furthermore, this unequal bias consequently will promote either a higher occurrence of green red or red green bigram sequences. It is worth taking a look at these correlations to see if they are strong enough to replace the previously stated test.

8. References

Bergman, A. S., Abercrombie, G., Spruit, S., Hovy, D., Dinan, E., Boureau, Y.-L., and Rieser, V. Guiding the Release of Safer E2E Conversational AI through Value Sensitive Design. In Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 39–52, Edinburgh, UK, September 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.sigdial-1.4

Kirchenbauer, J., Geiping, J., Wen, Y., Katz, J., Miers, I., & Goldstein, T. (2023, June 6). A watermark for large language models. [2301.10226] A Watermark for Large Language Models. https://arxiv.org/abs/2301.10226

Wouters, B. (2023, December 28). *Optimizing watermarks for large language models*. arXiv.org. https://arxiv.org/abs/2312.17295