Ensemble Watermarks for Large Language Models

Georg Niess[1], **Roman Kern**[1,2],

[1]Graz University of Technology, [2]Know Center Research GmbH,

Correspondence: georg.niess@tugraz.at

Abstract

As large language models (LLMs) reach human-like fluency, reliably distinguishing AI-generated text from human authorship becomes increasingly difficult. While watermarks already exist for LLMs, they often lack flexibility and struggle with attacks such as paraphrasing. To address these issues, we propose a multi-feature method for generating watermarks that combines multiple distinct watermark features into an ensemble watermark. Concretely, we combine acrostica and sensorimotor norms with the established red-green watermark to achieve a 98% detection rate. After a paraphrasing attack, the performance remains high with 95% detection rate. In comparison, the red-green feature alone as a baseline achieves a detection rate of 49% after paraphrasing. The evaluation of all feature combinations reveals that the ensemble of all three consistently has the highest detection rate across several LLMs and watermark strength settings. Due to the flexibility of combining features in the ensemble, various requirements and trade-offs can be addressed. Additionally, the same detection function can be used without adaptations for all ensemble configurations. This method is particularly of interest to facilitate accountability and prevent societal harm.

1 Introduction

The inception of transformer-based architectures (Vaswani et al., 2023) combined with large-scale pertaining (Devlin et al., 2019; Radford et al., 2019) has continuously improved the performance of modern large language models (LLMs). Humans increasingly struggle to distinguish between texts written by LLMs and those created by humans, as machine-generated texts sometimes even deceive people more often than human-written ones (Zellers et al., 2019). Although a growing range of post-hoc detectors exists, many detection methods that were once considered reliable for GPT-2 now struggle with GPT-3 and later versions (Fagni et al., 2021). This arms race between generation and detection intensifies as model size and capabilities continue to scale (Kaplan et al., 2020). Several potential cases of misuse are already associated with these advanced models (Ray, 2023).

Watermarks attempt to solve this by embedding a secret code into LLM output by modifying logits of the generated tokens during the generation process. However, as we will show later, a watermark with only a single feature has limited resilience against attacks like paraphrasing. To help against this weakness, we introduce an ensemble watermark that combines stylometric watermark features like acrostica and sensorimotor norms with the established red-green watermark feature introduced by Kirchenbauer et al. (2023). Our method is flexible and allows for a diverse set of features, and we draw inspiration from features used in the context of stylometry.

Stylometry is the statistical analysis of variations in literary style, with numerous stylometric features proposed in literature (Neal et al., 2018). These features serve as a writer's fingerprint and include aspects such as syntax, vocabulary, sentence structure, sentence length, and other unique authorial characteristics. All of these features can be statistically analyzed and utilized in tasks like authorship attribution (Stamatatos, 2009). However, traditional authorship attribution techniques have shown challenges when applied to even smaller language models like GPT-2 (Uchendu et al., 2020).

Sensorimotor norms are classifications based on human cognition. Perceptual modalities, such as "hearing", and action effectors, such as "hand", have been extensively researched in psychology and cognitive semantics (Lynott et al., 2020). However, there is still limited research in computer science on this topic. In our approach, a secret key derived from the generated output selects the sensorimotor category, thereby influencing the gener-

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2903–2916

ated words. For example, "smells funny" would be preferred over "looks funny" for the olfactory category.

An acrostic is a text in which the first letter of each sentence can be combined to spell out a hidden message or word. Historically, authors have used acrostics to encode their authorship (Johnson, 2006), often incorporating variations of their names into the hidden message. A notable example of an acrostic can be found in Appendix 4. In our method, the secret key determines which letters are used as the first letter of the first word in each generated sentence. The main contributions of this paper are as follows:

- We propose an ensemble watermark approach for LLMs based on changing token logits on a token and sentence-based level to embed novel stylometric features combined with an established red-green watermark.
- We show that this provides more resilience against paraphrasing attacks for three LLMs and three different parameter settings and provides the best detection rate.
- We propose a detection method that works for any combination of our ensemble watermark features, even in isolation, without any changes to the function.

The flexible nature of our ensemble watermark allows it to be adapted to different requirements while using the same detection method. All code and data generated is available on GitHub[1].

2 Background & Related Work

Several notable attempts have been made to detect machine-generated text without using language and stylometric features. Depending on their functional approach, these efforts can generally be categorized into two main groups.

2.1 Watermarking Approaches

Post-hoc detection. The techniques in this category aim to identify machine-generated text without adding watermarks or altering either the LLM itself or its output. Notable examples of methods in this include the classifier developed by OpenAI (Kirchner et al., 2023), GPTZero (Tian, 2023), and DetectGPT (Mitchell et al., 2023) which utilizes the probability curvature of text sampled from an LLM. DetectGPT relies on the principle that text typically exists within negative curvature regions of a model's log probability. This means that even minor changes to a sentence will decrease its log likelihood, as an LLM continuously aims to optimize the probability of each sentence. Additional post-hoc methods can be found in the survey by Jawahar et al. (2020). One advantage of post-hoc detection methods is that they can be applied to any suspected text without prior requirements. However, they are vulnerable to user attacks and become less effective with more complex language models (Chakraborty et al., 2023).

Watermarking and Red-Green Watermarks. Watermarking is the technique of hiding information in data that is difficult for others to remove but can be detected by an algorithm to read the hidden information. Some successful watermark implementations already exist, such as (Kirchenbauer et al., 2023), (Christ et al., 2023), and there is ongoing research by (Aaronson, 2022, 2023). The work by (Kirchenbauer et al., 2023), which we also include in our ensemble watermark and all our comparisons, divides tokens into two categories: a green list and a red list. Tokens in the green list are given a weight boost to enhance their representation in the generated output. As a result, the generated text predominantly consists of words from the green list, while human-written text naturally incorporates words from the red list. Additionally, Xiang et al. (2024) tackle the challenge of replacing sensitive words by utilizing Sentence-BERT (Reimers and Gurevych, 2019). Zhao et al. (2024) also use red-green watermarking, but instead of dynamically generating the red-green list, use a fixed list to simplify the grouping strategy. Kuditipudi et al. (2024) then test several sampling schemes to improve watermark performance. The Duwak approach (Zhu et al., 2024) combines the red-green watermark feature with a sampling scheme to improve watermarking; it does, however, not combine multiple logit watermark features as we do.

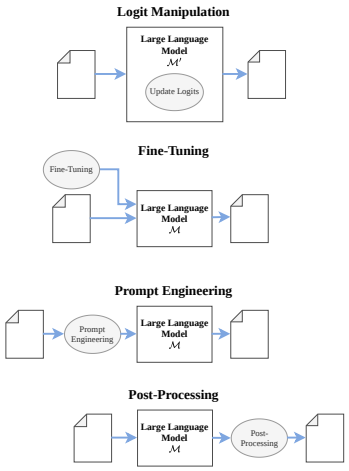
2.2 Stylometric Features

A wide variety of stylometric features have been proposed in the literature (Lagutina et al., 2019; Stamatatos, 2009). These features can be broadly classified into five categories: lexical, syntactic, semantic, structural, and domain-specific features. In Table 1, we have compiled a list of popular stylometric features found in the literature, along

[1] https://github.com/CommodoreEU/ensemble-watermark

Table 1: Overview of common stylometric features as identified in literature. Their suitability for four distinct ways to integrate watermarks into LLMs are reported based on a self-assessment. A black circle shows the best compatibility, followed by the semi-filled circle. For our ensemble we decided on sensorimotoric words and acrostics next to the established red-green feature, and on logit manipulation for watermarking. (Extended from Niess and Kern, 2024)

Feature Type	Logits	Fine-Tuning	Prompt	Post-Processing
N-Grams	●	●	○	○
Character Frequency	●	◐	◐	◐
Vocabulary Richness	●	●	◐	●
Word Distributions	◐	●	◐	◐
Word Length	●	●	◐	◐
Sentence Length	◐	●	○	○
Parts of Speech	○	●	○	○
Punctuation Frequency	◐	◐	○	◐
Sentence Complexity	◐	●	◐	◐
Synonyms	○	◐	○	●
Sensorimotoric Words	●	●	●	○
Acrostics	●	○	◐	●
Red-Green	●	○	○	◐



with additional watermark features we consider promising.

Sensorimotor Norms. Winter (2019) define sensory linguistics as the study of the relationship between senses and language. The Lancaster Sensorimotor Norms (Lynott et al., 2020) detail a set of 40,000 sensorimotor words, accompanied by a crowd-based assessment across 11 dimensions, which serve as the foundation for our sensorimotor features. Every word can be categorized into six perceptual groups: touch, hearing, smell, taste, vision, and interoception, as well as five action categories: mouth/throat, hand/arm, foot/leg, head (excluding mouth/throat), and torso. Khalid and Srinivasan (2022) utilize the Lancaster Norms and conclude that sensorial language is likely used intentionally and should not be viewed as a random phenomenon. Additionally, perceptual features have been suggested in the literature for authorship attribution, particularly in cross-language contexts (Bogdanova and Lazaridou, 2014).

Acrostica. Stein et al. (2014) explored the task of generating paraphrased versions of existing texts that also include acrostics. They approached this challenge as a search problem. Shen et al. (2019) employed a sequence-to-sequence network to generate texts with acrostics in both English and Chinese. More recently, using steganography for embedding secret messages in text has been investigated, utilizing BERT (Yi et al., 2022).

2.3 Watermarking Implementations

LLMs can utilize watermarks, including stylometric ones, in several ways. We present four different approaches in the following. They have different strengths and weaknesses. We outline four distinct approaches below, each with its own strengths and weaknesses. Additionally, we provide an overview of these methods and assess their expected effectiveness for different stylometric features in Table 1.

Logits Manipulation. In our approach, we have opted for directly manipulating the logits of the LLM tokens generated. Logits are the raw output values of a machine learning model before applying an activation function like softmax. They represent the unnormalized score or prediction for each token. This manipulation gives considerable control over how often single features are generated, though the difficulty lies in finding the correct logits that produce the desired features. An example of this type of watermark can be found in the work by Kirchenbauer et al. (2023).

Fine-tuning. A standard solution is to conduct additional training to change the output of an LLM. This method can also be applied to watermarks, although it is less controlled than other techniques. For instance, the written works of an author with a distinctive writing style can be used to fine-tune an LLM to produce text resembling that author's style (Li et al., 2023). However, a significant challenge is controlling what the model learns. The

LLM might not only pick up the desired watermark features but could also overfit to unrelated aspects, leading to a loss of generalization, domain mismatch, and decreased robustness in generating text.

Prompt Engineering. Another alternative is to use the inherent capabilities of LLMs by designing a system prompt that specifies the desired features (Zhou et al., 2023). For example, it is possible to tell contemporary LLMs to use only certain letters in their sentences (OpenAI ChatGPT, 2024). The primary challenges associated with this method include creating an effective prompt, protecting against users writing their own prompt, and understanding what prompts the language model can comprehend and adhere to. Generally, more powerful models are better suited for this approach.

Post-Processing. The final option is to process the text after generation is complete. This is how early attempts were made by Topkara et al. (2005) and Atallah et al. (2001) before LLMs were developed. It was commonly used to embed watermarks for copyright and document integrity purposes. Although the generative capabilities of LLMs have diminished the appeal of this approach, it can still be beneficial for implementing simple features like synonym replacement, which do not require generating new sentences.

3 Method

Our approach includes two main components: 1) a process for generating watermarks by manipulating logits with dynamic keys, and 2) a test procedure for detecting an existing watermark using statistical tests.

3.1 Watermark Generation

The generation algorithm modifies the logits of the language model during text generation to embed the features forming the watermark. It adjusts token probabilities to make certain words more likely, based on: The acrostic pattern, by boosting tokens that start with specific letters. Sensorimotor words, by boosting tokens associated with a target sensorimotor class. The red-green mechanism, by adjusting token probabilities based on a dynamically generated green list. We follow the procedure proposed in the original paper (Kirchenbauer et al., 2023) for splitting the red-green list. For the other features, a secret key is required, as described below.

Algorithm 1 Watermarked Text Generation

```
Initialize secret key: senso_class, acro_letter
Set δ_acro, δ_senso, δ_redgreen
while not done generating do
    Get current logits from the model
    if starting a new sentence then
        Adjust logits for acrostic boosting
    else
        Adjust logits for sensorimotor boosting
    end if
    Generate green list based on last token
    Adjust logits for red-green mechanism
    Sample next token from adjusted logits
    Update the secret key based on last word or
    sentence
end while
```

Secret Key Generation. A secret key is maintained throughout the generation process to control most of the features, i.e., it determines the sensorimotor class and the letter for the acrostic pattern. The key is updated based on the last word and the last sentence using secure hash functions. Both words and sentences are hashed using the same base function. Given a word w, the hash function maps it to an integer directly within a specified range $[a, b]$. Given a sentence s, we first lemmatize and remove stopwords and punctuation to get a sentence s'. The hash function is applied to s' to generate an integer within a range $[a', b']$.

$$\text{hash}(x) = \big(\text{int}\left(\text{SHA256}(x) \mod 2^{32}\right) \mod (b - a + 1)\big) + a$$

Logits Adjustment. During generation, the logits (raw scores before softmax) are adjusted to boost the probability of specific tokens. For example, if a new sentence is started, the initial token is boosted according to the target acrostic letter. For each token t:

$$\text{logits}[t] += \delta_{\text{acro}} \cdot \mathbf{1}\{\text{starts\_with\_acrostic\_letter}\}$$

Otherwise, boost tokens that are associated with the current sensorimotor class. For each token t:

$$\text{logits}[t] += \delta_{\text{senso}} \cdot \mathbf{1}\{\text{token\_in\_sensorimotor\_class}\}$$

The red-green mechanism is based on the work by (Kirchenbauer et al., 2023). A green list is generated based on the last token t_{last}. A random number is then seeded by a generator with:

seed $=$ hash(t_{last}). This is used to generate a random permutation of the vocabulary, where the first $\gamma \cdot V$ tokens are selected as the green list, where V is the vocabulary size and γ is a predefined proportion (e.g., 0.5) For tokens in the green list: logits$[t] += \delta_{\text{redgreen}}$

It is important to ensure consistent tokenization between generation and detection and appropriately handle special tokens (e.g., BOS, EOS). Streaming generation is used to update the secret key and adjust logits at each step during generation. To detect sentence boundaries, punctuations are used (e.g., ., !, ?).

We decided to assign a relatively large weight boost to acrostica because the beginning of the sentence is more flexible, allowing the rest of the sentence to adapt to this change easily. In contrast, we chose a small weight for sensorimotor words to prevent the model from becoming overly biased.

All features have in common that their strength is controlled via a δ parameter. In the evaluation we study the impact of these parameters on the generated text. While we opted for fixed values for the weights, they could also be chosen dynamically, depending on how long it has been since a desired feature was chosen or on the distribution of the current weights.

3.2 Watermark Detection

The detection of the watermark works similarly to that of the generation. A secret key is maintained the same way, but instead of modifying logits, the generated token is compared based on the key, and the probability of that token occurring is calculated.

The probability of detecting an acrostic watermark is calculated using the formula:

$$P_{\text{acrostic}} = 1 - \sum_{i=0}^{k-1} \binom{n}{i} p^i (1-p)^{n-i}$$

where n is the number of acrostic checks (total sentences minus one), k is the number of correct matches observed, and $p = \frac{1}{26}$ represents the probability of a random match.

The probability of detecting a sensorimotor watermark uses class-specific frequencies from the Google N-gram corpus or evenly split probabilities. For each class c, the baseline probability p_c is calculated. Given k_c matches out of n_c words for class c, the probability is:

$$P_{\text{sensorimotor},c} = 1 - \sum_{i=0}^{k_c-1} \binom{n_c}{i} p_c^i (1-p_c)^{n_c-i}$$

Algorithm 2 Watermark Detection Algorithm

Input: Text T
Initialize: Load sensorimotor norms and class frequencies from the corpus
Initialize secret key: sensorimotor_class, acrostic_letter
Initialize counters for acrostic matches k, total checks n
Initialize counters for sensorimotor matches per class k_c, total words per class n_c
Initialize variables for red-green detection: total transitions T, green tokens G
Split text T into sentences $S = [s_1, s_2, \ldots, s_N]$
for each sentence s_i in S **do**
  Split s_i into words $W = [w_1, w_2, \ldots, w_m]$
  if $i > 1$ **then**
    Acrostic Check: Compute expected letter using hash_sentence(s_{i-1})
    Compare with first letter of w_1
    Increment k if match found
  end if
  for each word w_j in W **do**
    Update sensorimotor class using hash_word(w_{j-1})
    $c \leftarrow$ sensorimotor_class
    if w_j is in sensorimotor dictionary **then**
      $n_c \leftarrow n_c + 1$
      if w_j belongs to class c **then**
        $k_c \leftarrow k_c + 1$
      end if
    end if
    Red-Green Detection: Update green list based on previous token
    Increment G if w_j is in green list
  end for
end for
Calculate Probabilities: Compute acrostic probability P_{acrostic}
Compute sensorimotor probability $P_{\text{sensorimotor}} = \prod_c P_{\text{sensorimotor},c}$
Compute red-green probability P_{redgreen}
Compute Final Score:

$$\text{final\_score} = P_{\text{acrostic}} \times P_{\text{sensorimotor}} \times P_{\text{redgreen}}$$

Output: final_score

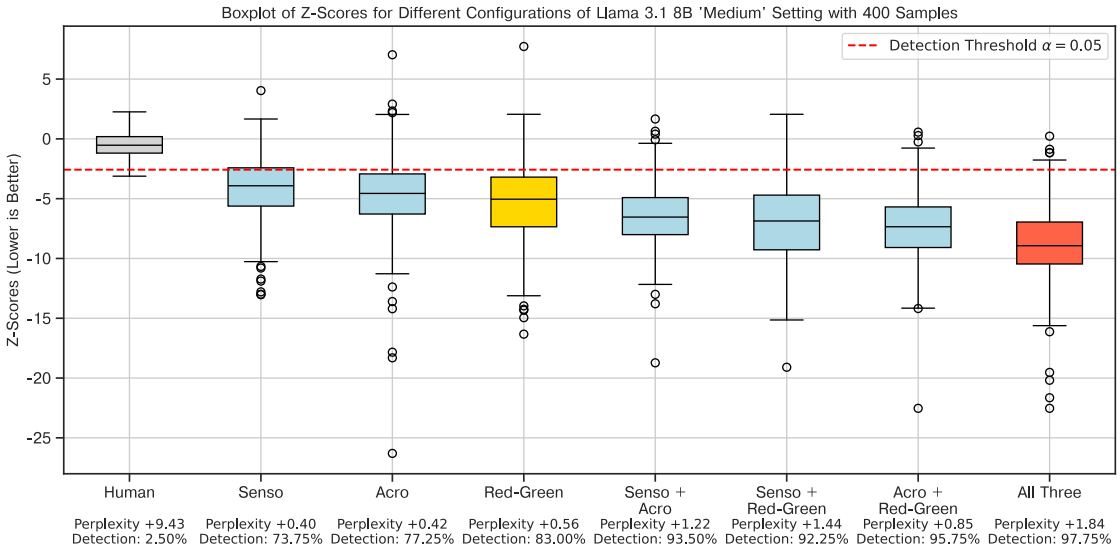


Figure 1: Plot of the Z-Scores of different configurations of the watermark ensemble (Llama 3.1 8B, medium strength watermark). The human and red-green results serve as baselines. Each watermark configuration has 400 unique samples. The combination of all features achieves the best detection rate of 97.75%. The combination of acrostic and red-green might be attractive for its more moderate increase in perplexity (+0.85), while still achieving a detection rate of 95.75%. The red-green baseline achieves a detection rate of 83.00%.

The combined probability across all classes is:

$$P_{\text{sensorimotor}} = \prod_{c} P_{\text{sensorimotor},c}$$

The probability calculation for the red-green watermark involves the following parameters: T, which represents the total number of transitions; G, the number of tokens in the green list observed; and $\gamma = 0.5$, the probability parameter. The expected number E and variance Var are:

$$E = \gamma T, \quad \text{Var} = T\gamma(1-\gamma)$$

The Z-score and probability is calculated as:

$$Z = \frac{G - E}{\sqrt{\text{Var}}}, \quad P_{\text{redgreen}} = 1 - \Phi(Z)$$

where Φ is the cumulative distribution function of the standard normal distribution. Assuming independence, the final score is:

$$\text{final\_score} = P_{\text{acrostic}} \times P_{\text{sensorimotor}} \times P_{\text{redgreen}}$$

4 Evaluation

In this section, we describe the experiments conducted, assess the resilience against a paraphrasing attack, and conduct an ablation study to analyze the impact of text length.

4.1 Implementation Details.

We used three LLMs, Llama 3.1 8B (Dubey et al., 2024), Llama 3.2 3B, and Mistral 7B (Jiang et al., 2023). We utilized the logits processor of Hugging Face (Wolf et al., 2019) to manipulate the generation. To create prompts with human baselines, we randomly select texts from the C4 RealNewsLike dataset (Raffel et al., 2019), trim a fixed length of tokens as "baseline" completions from the end, and treat the remaining tokens as the prompt.

We create a weak, middle, and strong parameter setting to test different parameters for the features. The medium setting $[\delta_{senso} = 2.5, \delta_{acro} = 20.0, \delta_{redgreen} = 2.0]$ has 400 samples per configuration, while the strong $[\delta_{senso} = 5.0, \delta_{acro} = 40.0, \delta_{redgreen} = 10.0]$ and weak $[\delta_{senso} = 1.0, \delta_{acro} = 10.0, \delta_{redgreen} = 1.0]$ setting contain 300 samples per configuration. We use a level of $\alpha = 0.05$ to determine the statistical significance that a watermark can be recovered. This allows us to calculate a detection rate showing how many samples would have been detected as a watermark with this α threshold.

4.2 Results

Figure 1 provides an overview of combinations of features for Llama 3.1 8B, with a lower Z-Score indicating improved detectability. Overall, combi-

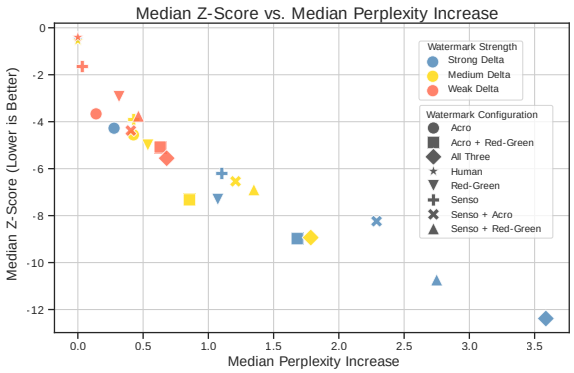


Figure 2: Llama 3.1 8B tested with three different parameter settings. Higher parameter values for the δ weights lead to lower Z-Scores (improved detectability) and an almost linear increase in the perplexity. The acrostic feature stands out for having the least impact on perplexity, with the ensemble of all features providing the best detectability for each parameter setting.

Figure 3: Heatmap of adjusted p-values from pairwise Mann-Whitney U tests between configurations (Bonferroni correction applied) for Llama 3.1 8B with the medium parameter setting. Lower adjusted p-values indicate a significant difference after correction for multiple comparisons.

nations of watermark features have a higher detection rate than single features. Of the single features, the red-green feature provides the best detectability. The combination of all features has the best detection rate, with an increase of 17.78% compared to the red-green feature. We also tested for statistical significance in Figure 3. This finding is consistent across all experiments we conducted and reported in Appendix A for all settings.

The scatter plot in Figure 2 additionally shows the three considered parameter settings (weak, medium, strong), together with the increase in perplexity. Higher perplexity reflects more substantial deviations in the output distribution. For all parameter settings, combining all features provides the best detectability but with an associated increase in perplexity. Interestingly, the acrostic feature is associated with the smallest increase in perplexity, i.e., the most minor influence on the distribution of overall tokens.

Experiments on Paraphrasing Attacks. We conducted experiments to assess the vulnerability of the various watermarking configurations via paraphrasing attacks. This experiment involves tokenizing the watermarked text, then iteratively replacing one word with a `<mask>`, and using T5 (Raffel et al., 2019) to generate candidate replacement sequences via beam search. If one candidate differs from the original string, the attack succeeds, replacing the span with the new text. This is repeated until at least 10% of the watermarked text is replaced. The results can be found in Table 2 and show that the combination of all three features has the highest detection rate after the paraphrasing attack for all of the considered LLMs and parameter settings. Llama 3.1 3B with the strong parameter setting retains over 95% detection rate, in contrast to the red-green feature in isolation, which drops to 49%. Here, the sensorimotor feature stands out, being more resilient to this type of attack retaining over 80% detection rate. The acrostic feature alone does not appear to be resilient enough to the attack and only achieves a low detection rate. However, it still contributes to overall watermark resilience, which can be seen in the improvement of the ensemble of all features against using only the combination of sensorimotor and red-green feature. Overall, using multiple features together contributes to much greater resilience against paraphrasing attacks.

4.3 Ablation Study Text Length

To further analyze the influence of text length on the contribution of each feature, we do incremental sentence partitioning, where each version represents a progressively shorter subset of each text, starting with the total sentence count and reducing by one until reaching a single sentence. In Figure 4, the detection rate of each feature can be seen for three different watermark configurations. While acrostics are not as sensitive to differences in the δs, both sensorimotor and red-green features are associated with a jump in detection rate once an

Configuration	Human	Senso	Acro	Red-Green	Senso + Acro	Senso + Red-Green	Acro + Red-Green	All Three
Llama 3.1 8B (Strong)	0.34	80.41	28.52	49.14	89.35	84.19	55.67	**95.19**
Llama 3.2 3B (Strong)	0.97	85.11	31.39	54.05	90.61	90.29	64.08	**95.79**
Llama 3.1 8B (Medium)	2.42	58.06	28.23	34.14	73.92	70.70	46.77	**82.53**
Mistral 7B (Medium)	1.44	54.87	42.60	23.47	69.31	58.84	40.07	**73.65**
Llama 3.2 3B (Weak)	1.32	26.07	27.72	29.04	38.61	43.56	35.97	**44.88**

Table 2: Detection rate of different LLM configurations after a paraphrasing attack changing at least 10% of each text. All three watermark features combined are consistently the best configuration. Interestingly, Mistral 7B has better results for the acrostic feature since it tended to produce more numerous shorter sentences.

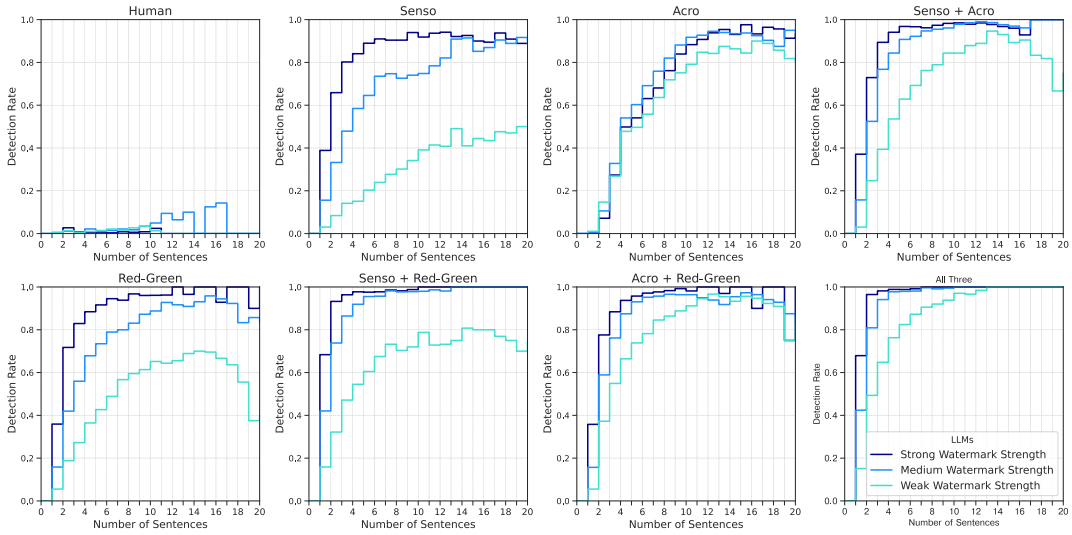


Figure 4: For the ablation study, we study the influence of the number of output sentences on the detection rate. We selected Llama 3.1 8B as language model, and present results for the weak, middle and strong parameter settings. The ensemble of all three features provides the best overall detection rate with the fewest sentences.

increase in the weights is applied to the logits. It can also be seen that a combination of all three features reaches a high detection rate earlier than other configurations and that the weak setting even beats some of the medium-strength configurations, hinting that for specific requirements and trade-offs specific combinations in the ensemble are best suited.

5 Conclusion

We introduced a novel ensemble watermarking method for large language models, specifically generative transformer models. Among the various approaches for integrating watermarks into LLMs, we chose to manipulate the probabilities directly when generating tokens. Following existing work, we generate a key dynamically derived directly from the generated text to control the watermark features. Out of many possible stylometric features, we focused on two features for our evaluation, the acrostic and sensorimotor norms, and combined them with an established method from literature, red-green lists.

For the experiment, we selected three different LLMs and three levels of watermark strength. In the evaluation, we found that each of the three features has different characteristics in relation to the detectability and perplexity of the generated sentences.

The acrostic feature has the most negligible impact on perplexity, but its performance depends on sentence number and length and has less resilience against paraphrasing than other features. The sensorimotor feature is similar in watermark performance and perplexity impact as the established red-green feature but shows increased resilience against paraphrasing. On the other hand, the red-green feature shows a balanced performance and combines well with the other (stylometric) features.

Overall, we propose a flexible and resilient en-

semble watermark for text that works with short text lengths and does not require a different sampling strategy, expensive additional model training, or an LLM for testing. Since our method allows for many types of key generation and (stylometric) features, an exploration of further combinations, including watermark sampling strategies, will be part of the future work.

6 Limitations and Risks

Currently, stylometric watermarks are only implemented for the English language. In principle, our watermark can be applied to almost all languages with two restrictions based on the stylometric features we use. Acrostica require the language to have an alphabet similar in size to the Latin alphabet, so Chinese or Japanese characters would make this aspect nonfunctional. As for sensorimotor norms, all languages share this aspect since they relate directly to how our brains work (Connell et al., 2012). A different database would be ideal for each language, although the English one by Lynott et al. (2020) could also be translated for the same effect, since sensorimotor accuracy is not important for watermarking purposes.

This also leads to the topic of attacks. While paraphrasing attacks are commonly considered in literature, other attacks are also plausible.

Another possible limitation is as to what language models are compatible to this method. Currently, we have only shown compatibility to the popular decoder models Llama 3.1/3.2 and Mistral. However, the only architecture limitation for a language model to be compatible is to have a logits layer to manipulate generation and for the model to be sequential to be able to generate new keys.

Risks. Our work contributes to trustworthy AI, in particular by addressing the topic of accountability for proprietary language models. Since the key generation can be made specific for each LLM, this allows to link generated text back to its origin. Thus, the proposed watermark and its methods have high potential to help reduce the negative risks on society by the harmful use of LLMs.

References

Scott Aaronson. 2022. My AI safety lecture for UT effective altruism.

Scott Aaronson. 2023. Watermarking of large language models.

Mikhail J. Atallah, Victor Raskin, Michael Crogan, Christian Hempelmann, Florian Kerschbaum, Dina Mohamed, and Sanket Naik. 2001. Natural language watermarking: design, analysis, and a proof-of-concept implementation. In Gerhard Goos, Juris Hartmanis, Jan Van Leeuwen, and Ira S. Moskowitz, editors, *Information Hiding*, Lecture Notes in Computer Science, pages 185–200. Springer, Berlin, Heidelberg.

Dasha Bogdanova and Angeliki Lazaridou. 2014. Cross-language authorship attribution. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 2015–2020, Reykjavik, Iceland. European Language Resources Association (ELRA).

Souradip Chakraborty, Amrit Singh Bedi, Sicheng Zhu, Bang An, Dinesh Manocha, and Furong Huang. 2023. On the possibilities of AI-generated text detection. *arXiv preprint*. ArXiv:2304.04736 [cs].

Miranda Christ, Sam Gunn, and Or Zamir. 2023. Undetectable watermarks for language models. Report Number: 763.

Louise Connell, Dermot Lynott, and Felix Dreyer. 2012. A functional role for modality-specific perceptual systems in conceptual representations. *PLOS One*, 7(3):e33321.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [No title found]. In *Proceedings of the 2019 Conference of the North*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Tiziano Fagni, Fabrizio Falchi, Margherita Gambini, Antonio Martella, and Maurizio Tesconi. 2021. TweepFake: about detecting deepfake tweets. *PLOS One*, 16(5):e0251415. ArXiv:2008.00036 [cs].

Ganesh Jawahar, Muhammad Abdul-Mageed, and Laks Lakshmanan, V.S. 2020. Automatic detection of machine generated text: a critical survey. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2296–2309, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7B. *arXiv preprint*. ArXiv:2310.06825 [cs].

Ian Johnson. 2006. Authorial self-identification in the acrostics of walton's" boethius" and the question of john bonejohn. *Carmina Philosophiae*, 15:1–12.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint*. ArXiv:2001.08361 [cs, stat].

Osama Khalid and Padmini Srinivasan. 2022. Smells like teen spirit: An exploration of sensorial style in literary genres. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 55–64.

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. arXiv. ArXiv:2301.10226 [cs] shortConferenceName: ICML.

Jan Hendrik Kirchner, Jan Ahmad, Scott Aaronson, and Jan Leike. 2023. New AI classifier for indicating AI-written text.

Rohith Kuditipudi, John Thickstun, Tatsunori Hashimoto, and Percy Liang. 2024. Robust distortion-free watermarks for language models. *Trans. Mach. Learn. Res.*, 2024.

Ksenia Lagutina, Nadezhda Lagutina, Elena Boychuk, Inna Vorontsova, Elena Shliakhtina, Olga Belyaeva, Ilya Paramonov, and P.G. Demidov. 2019. A survey on stylometric text features. In *2019 25th Conference of Open Innovations Association (FRUCT)*, pages 184–195, Helsinki, Finland. IEEE. ISSN: 2305-7254.

Cheng Li, Mingyang Zhang, Qiaozhu Mei, Yaqing Wang, Spurthi Amba Hombaiah, Yi Liang, and Michael Bendersky. 2023. Teach LLMs to personalize - an approach inspired by writing education. *arXiv.org*, abs/2308.7968. ArXiv:2308.07968 [cs].

Dermot Lynott, Louise Connell, Marc Brysbaert, James Brand, and James Carney. 2020. The lancaster sensorimotor norms: multidimensional measures of perceptual and action strength for 40,000 english words. *Behavior Research Methods*, 52(3):1271–1291.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D. Manning, and Chelsea Finn. 2023. DetectGPT: zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *ICML'23*, pages 24950–24962, Honolulu, Hawaii, USA. JMLR.org.

Tempestt Neal, Kalaivani Sundararajan, Aneez Fatima, Yiming Yan, Yingfei Xiang, and Damon Woodard. 2018. Surveying stylometry techniques and applications. *ACM Computing Surveys*, 50(6):1–36.

Georg Niess and Roman Kern. 2024. Stylometric watermarks for large language models. *Preprint*, arXiv:2405.08400.

OpenAI ChatGPT. 2024. Ethereal poem in e. Feb 1 Version.

Alec Radford, Jeff Wu, Rewon Child, D. Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*.

Partha Pratim Ray. 2023. ChatGPT: a comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope. *Internet of Things and Cyber-Physical Systems*, 3:121–154.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Liang-Hsin Shen, Pei-Lun Tai, Chao-Chung Wu, and Shou-De Lin. 2019. Controlling sequence-to-sequence models-a demonstration on neural-based acrostic generator. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 43–48.

Efstathios Stamatatos. 2009. A survey of modern authorship attribution methods. *Journal of The American Society for Information Science and Technology*, 60(3):538–556.

Benno Stein, Matthias Hagen, and Christof Bräutigam. 2014. Generating acrostics via paraphrasing and heuristic search. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2018–2029.

Edward Tian. 2023. GPTZero | the trusted AI detector for ChatGPT, GPT-4, & more.

Mercan Topkara, Cuneyt M. Taskiran, and Edward J. Delp Iii. 2005. Natural language watermarking. page 441, San Jose, CA.

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, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *arXiv preprint*. ArXiv:1706.03762 [cs].

Bodo Winter. 2019. Sensory linguistics. *Sensory Linguistics*, pages 1–303.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. HuggingFace's transformers: state-of-the-art natural language processing. *arXiv.org*, abs/1910.3771. ArXiv:1910.03771 [cs].

Lingyun Xiang, Yangfan Liu, and Zhongliang Yang. 2024. A reversible natural language watermarking for sensitive information protection. *Information Processing & Management*, 61(3):103661.

Biao Yi, Hanzhou Wu, Guorui Feng, and Xinpeng Zhang. 2022. Alisa: Acrostic linguistic steganography based on bert and gibbs sampling. *IEEE Signal Processing Letters*, 29:687–691.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Xuandong Zhao, Prabhanjan Vijendra Ananth, Lei Li, and Yu-Xiang Wang. 2024. Provable robust watermarking for AI-generated text. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

Yongchao Zhou, Andrei Ioan Muresanu, Ziwen Han, Keiran Paster, Silviu Pitis, Harris Chan, and Jimmy Ba. 2023. Large language models are human-level prompt engineers. *arXiv preprint*. ArXiv:2211.01910 [cs].

Chaoyi Zhu, Jeroen Galjaard, Pin-Yu Chen, and Lydia Chen. 2024. Duwak: dual watermarks in large language models. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 11416–11436, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

A Appendix

Configuration	p-Senso	p-Acro	p-Red-Green
Human	0.60140883	0.99999999	0.94382440
Senso	0.00011047	0.99999999	0.88591727
Acro	0.63667415	0.00000931	0.94849433
Red-Green	0.63739815	0.99999999	0.00000102
Senso + Acro	0.00001441	0.00004255	0.91098964
Senso + Red-Green	0.00034610	0.99999999	0.00000011
Acro + Red-Green	0.58650698	0.00022102	0.00000004
All Three	0.00002030	0.00022102	0.00000001

Table 3: Table of p-values of each feature under different ensemble settings. All values are from Llama 8B with the medium strength watermark setting.

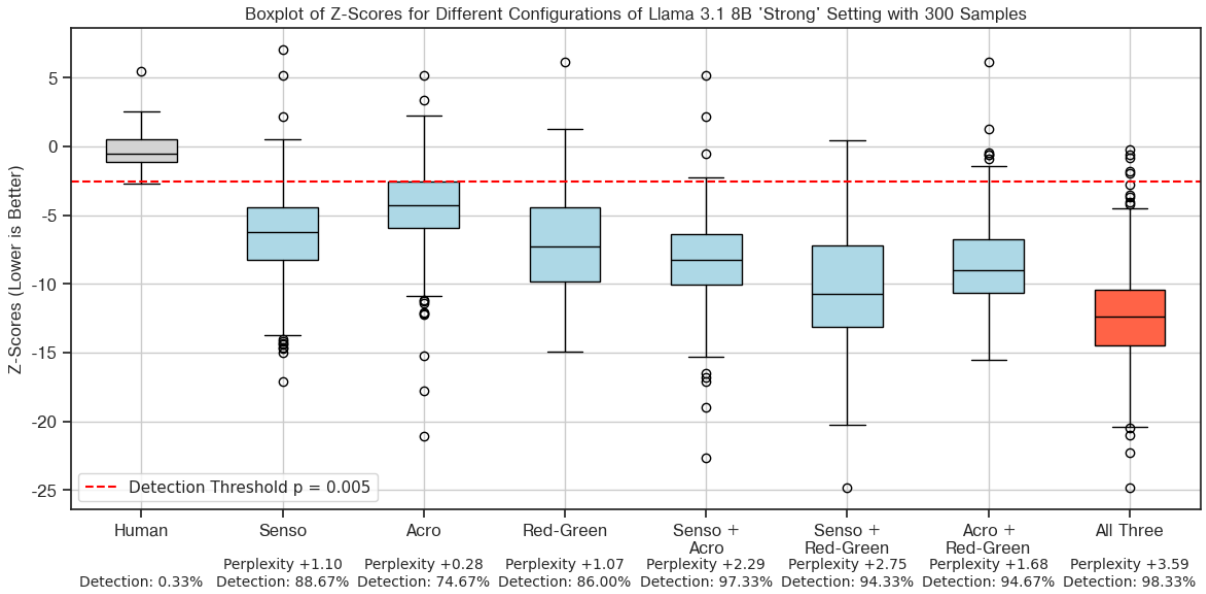


Table 4: Example of an acrostic written by Arnold Schwarzenegger in a letter to the members of the California State Assembly, which has been used in literature for illustration before (Stein et al., 2014).

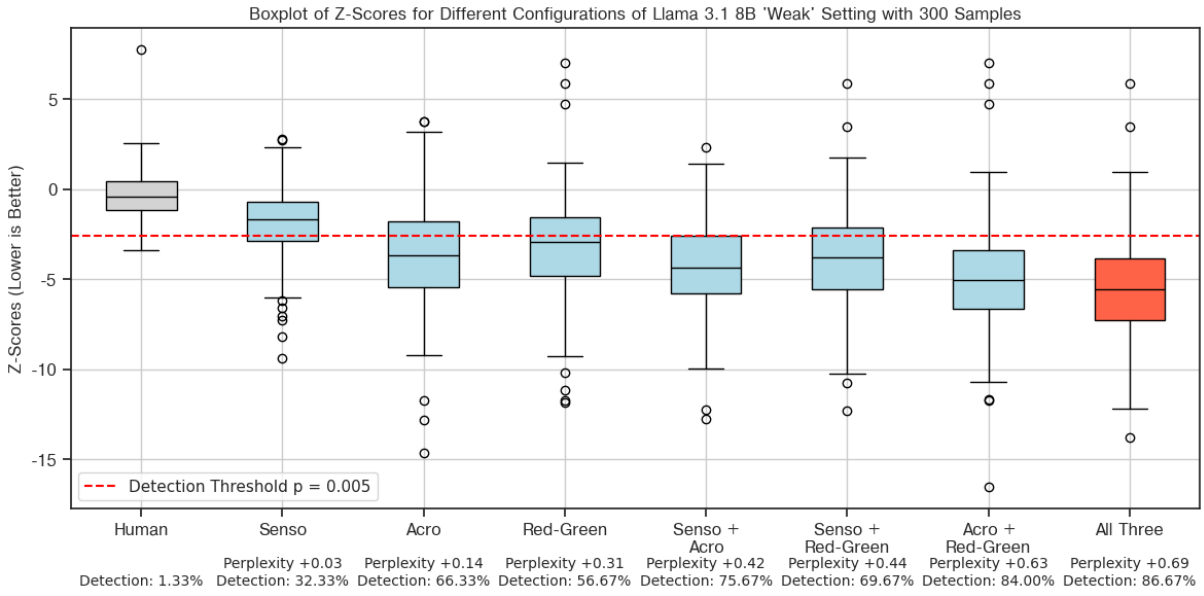
To the Members of the California State Assembly:

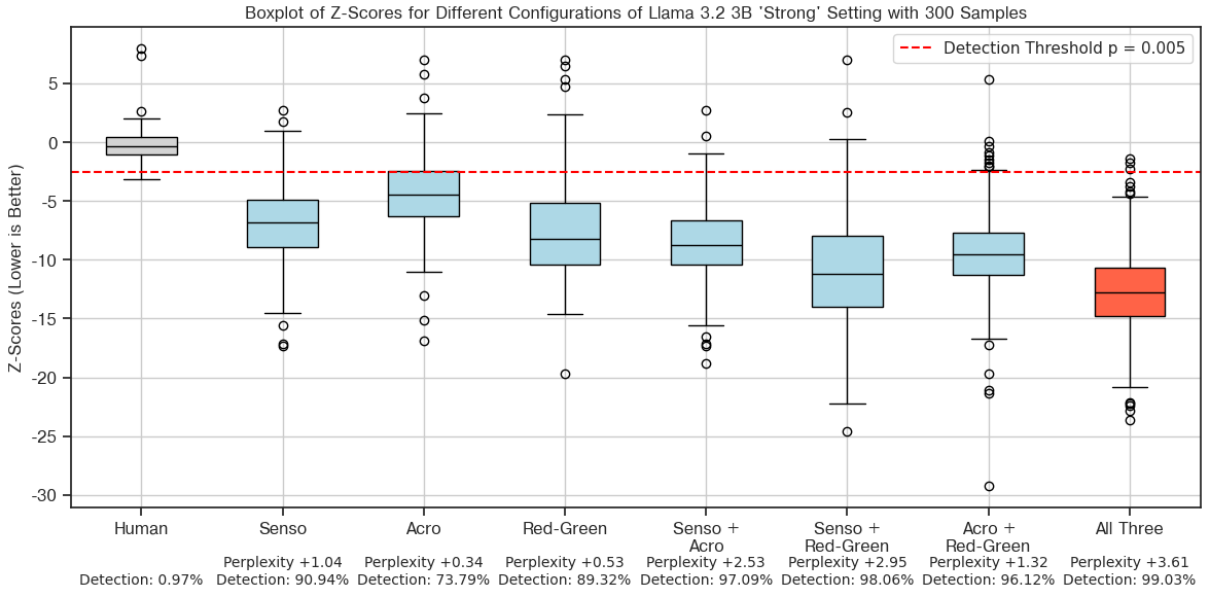
I am returning Assembly Bill 1176 without my signature.

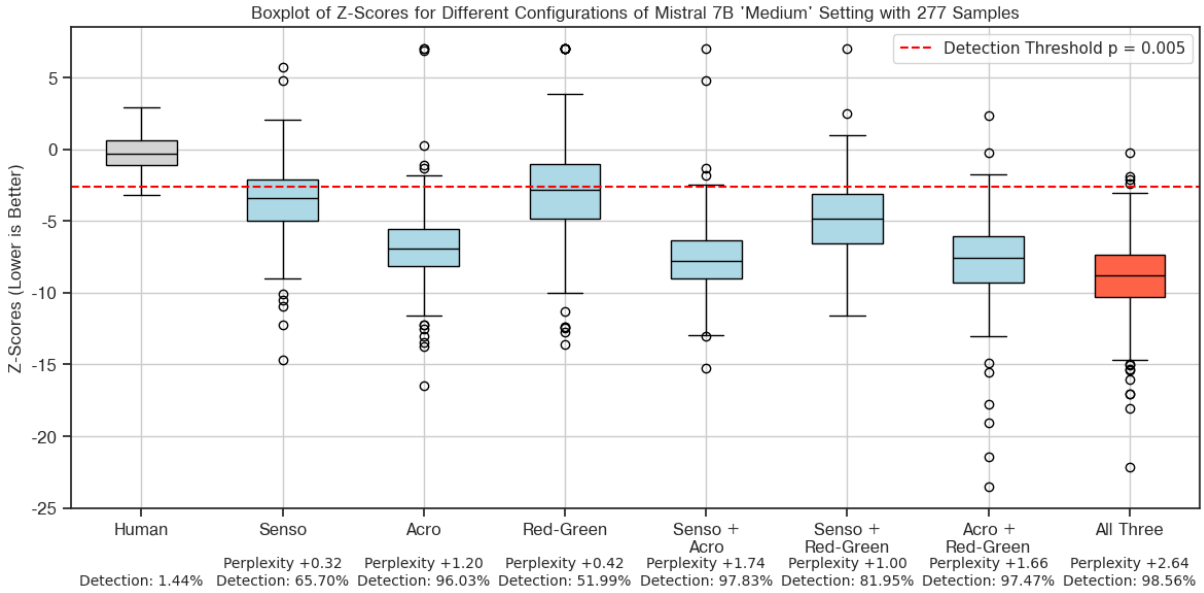
F or some time now I have lamented the fact that major issues are overlooked while many
u nnecessary bills come to me for consideration. Water reform, prison reform, and health
c are are major issues my Administration has brought to the table, but the Legislature just
k icks the can down the alley.

Y et another legislative year has come and gone without the major reforms Californians
o verwhelmingly deserve. In light of this, and after careful consideration, I believe it is
u nnecessary to sign this measure at this time.

Sincerely,
Arnold Schwarzenegger


Boxplot of Z-Scores for Different Configurations of Llama 3.1 8B 'Weak' Setting with 300 Samples
Z-Scores (Lower is Better)
5
0
-5
-10
-15
Detection Threshold p = 0.005
Human
Detection: 1.33%
Senso
Perplexity +0.03
Detection: 32.33%
Acro
Perplexity +0.14
Detection: 66.33%
Red-Green
Perplexity +0.31
Detection: 56.67%
Senso + Acro
Perplexity +0.42
Detection: 75.67%
Senso + Red-Green
Perplexity +0.44
Detection: 69.67%
Acro + Red-Green
Perplexity +0.63
Detection: 84.00%
All Three
Perplexity +0.69
Detection: 86.67%


Boxplot of Z-Scores for Different Configurations of Llama 3.2 3B 'Strong' Setting with 300 Samples
Z-Scores (Lower is Better)
Detection Threshold p = 0.005
5
0
-5
-10
-15
-20
-25
-30
Human
Detection: 0.97%
Senso
Perplexity +1.04
Detection: 90.94%
Acro
Perplexity +0.34
Detection: 73.79%
Red-Green
Perplexity +0.53
Detection: 89.32%
Senso + Acro
Perplexity +2.53
Detection: 97.09%
Senso + Red-Green
Perplexity +2.95
Detection: 98.06%
Acro + Red-Green
Perplexity +1.32
Detection: 96.12%
All Three
Perplexity +3.61
Detection: 99.03%


Boxplot of Z-Scores for Different Configurations of Mistral 7B 'Medium' Setting with 277 Samples
Detection Threshold p = 0.005
Z-Scores (Lower is Better)
5
0
-5
-10
-15
-20
-25
Human
Detection: 1.44%
Senso
Perplexity +0.32
Detection: 65.70%
Acro
Perplexity +1.20
Detection: 96.03%
Red-Green
Perplexity +0.42
Detection: 51.99%
Senso + Acro
Perplexity +1.74
Detection: 97.83%
Senso + Red-Green
Perplexity +1.00
Detection: 81.95%
Acro + Red-Green
Perplexity +1.66
Detection: 97.47%
All Three
Perplexity +2.64
Detection: 98.56%