Three Bricks to Consolidate Watermarks for Large Language Models

Pierre Fernandez[1,3], Antoine Chaffin[1,2], Karim Tit[1], Vivien Chappelier[2], Teddy Furon[1]

[1]*Centre Inria de l'Université de Rennes*   [2]*Imatag*   [3]*FAIR, Meta*

Abstract—**Discerning between generated and natural texts is increasingly challenging. In this context, watermarking emerges as a promising technique for ascribing text to a specific generative model. It alters the sampling generation process to leave an invisible trace in the output, facilitating later detection. This research consolidates watermarks for large language models based on three theoretical and empirical considerations. First, we introduce new statistical tests that offer robust theoretical guarantees which remain valid even at low false-positive rates (less than 10^{-6}). Second, we compare the effectiveness of watermarks using classical benchmarks in the field of natural language processing, gaining insights into their real-world applicability. Third, we develop advanced detection schemes for scenarios where access to the LLM is available, as well as multi-bit watermarking.**

Index Terms—**Watermarking, Large Language Model**

I. INTRODUCTION

The misuse of Large Language Models (LLMs) like Chat-GPT [1], Claude [2], or the open-sourced LLaMA [3] may become a threat as their availability and capabilities expand [4]–[6]. LLMs might help generate fake news by reducing costs to spread disinformation at scale [7], [8], with a potential impact on public opinion and democratic outcomes [9]. They could help impersonate people, facilitate scams [10], or make student assessments impossible. Enforcing fair and responsible usage through regulations and technical means would be useful.

Monitoring the usage of LLMs with passive forensics is difficult because generated texts are hardly distinguishable from real ones, be it for humans or algorithms [11], [12]. Watermarking is a promising technique explored for generative image models [13]–[15] and generative text LLMs [16]–[19]. In this case, watermarking either alters the sample generation process [16], [19] or changes the probability distribution of the generated tokens [17], [20], to leave an imperceptible trace in the generated text. This literature then describes a detection mechanism analyzing the generated tokens to see if their distribution follows the one induced by the watermark.

We introduce three contributions to consolidate the current literature, one for each of the following paragraphs and sections. Each part can be read independently.

First, false positives can have serious consequences in contexts where the integrity and accuracy of results are essential, such as falsely accusing a user of producing fake news or a student of cheating in an exam. However, current approaches [17], [18] focus their study on sensitivity (True Positive Rate: TPR)
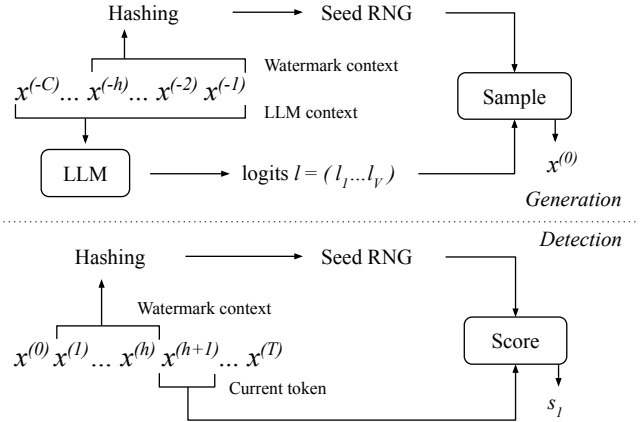
Fig. 1: General illustration of watermarking for LLM (top: generation, bottom: detection). Details and notations in Sect. II-B.

rather than on specificity (linked to False Positive Rate: FPR). The FPR has never been empirically checked at interesting scales (with more than 1k negative examples). Our large-scale experiments reveal that hypotheses of previous works do not hold and that their detection thresholds largely underestimate the false positives at low FPR. This work provides grounded statistical tests that theoretically guarantee false positive-rates and accurate p-values in real-world regimes. We validate them empirically and show that they provide a close-to-perfect control of the FPR, even at low values ($< 10^{-6}$).

Second, we compare the watermarking methods, analyzing practical implications of watermarks on traditional Natural Language Processing (NLP) benchmarks. Indeed, current watermark evaluation mainly considers the deviation from the original LLM distribution, for example using perplexity. This is in contrast with the LLM litterature, where models are rather evaluated on their effective usefulness, *e.g.* free-form completion tasks such as question answering. Such evaluations are much more informative on the actual abilities of the model when used on downstream tasks.

Third, we expand these algorithms to advanced detection schemes. When access to the LLM is possible at detection time, we provide optimal statistical tests. We also investigate multi-bit watermarking (hiding binary messages as watermarks) when current approaches only tackle zero-bit watermarking. This allows not only to determine whether the text was generated by the watermarked LLM, but also to identify which version of the model generated it.

II. TECHNICAL BACKGROUND

A. Large Language Models (LLMs)

LLMs are neural networks that generate text by computing the likelihood of generating a sequence of tokens given a context [21]. This paper focuses on decoder-only models, a.k.a. auto-regressive LLMs. The tokens are pieces of words from a vocabulary $\mathcal{V}$. From a context $x^{(-C)}, ..., x^{(-1)}$, the model estimates the probability of each token of $\mathcal{V}$ being the next. It computes a vector $\ell \in \mathbb{R}^{|\mathcal{V}|}$ of logits, transformed into

$$\left(\mathbb{P}\left(X^{(0)} = x \middle| x^{(-C)}, \dots, x^{(-1)} \right) \right)_{x \in \mathcal{V}} = \text{softmax}(\ell; \theta) \quad (1)$$

where θ is a temperature. The generation of a sequence from the context samples a token from this distribution, then appends it to the context and iterates the process. Various sampling schemes exist: greedy search, beam search, top-k sampling [22], [23], nucleus-sampling (top-p) [24], etc.

B. Watermarking Text Generation

1) Modification of the Distribution [17], [18], [20]: The original distribution (1), denoted $\mathbf{p}$ for short, is replaced by a similar distribution $\mathbf{q} = F(\mathbf{p}, k)$ where k is a secret key and F an arbitrary function. In the work of Kirchenbauer *et al.* [17], the secret key determines a partitioning of $\mathcal{V} = \mathcal{G}_k \cup \bar{\mathcal{G}}_k$. The greenlist $\mathcal{G}_k$ contains $\gamma|\mathcal{V}|$ tokens, where $\gamma \in [0, 1]$. The logit of every token in the greenlist is incremented by $\delta > 0$, and the softmax operator outputs $\mathbf{q}$. The sampling then proceeds as usual. Intuitively, this increases the probabilty of generating greenlist tokens. On the other hand, $\mathbb{E}[F(\mathbf{p}, K)] = \mathbf{p}$ so on expectation over the set of cryptographic keys, watermarking does not bias the global distribution of words (K being the random variable representing the key).

The detection tokenizes the text and counts how many tokens are in their greenlist. More formally, for a text of T tokens, the score S_T is the number of greenlist tokens ($x^{(t)}$ and $k^{(t)}$ respectively indicate the tth token and key):

$$S_T = \sum_{t=1}^{T} \mathbb{1}\left(x^{(t)} \in \mathcal{G}_{k^{(t)}} \right). \quad (2)$$

2) Modification of the Sampling [16], [19]: The watermark embedding replaces the traditional sampling schemes by a deterministic process. For instance, Aaronson *et al.* [16] choose the next token by computing $x^{(0)} = \arg\max_{v \in \mathcal{V}} \mathbf{r}_v^{1/\mathbf{p}_v}$, where $\mathbf{p}$[1] is the distribution (1) and $\mathbf{r} \in [0,1]^{|\mathcal{V}|}$ a secret vector generated from the secret key k. Intuitively, this encourages the generation of tokens that have both high $\mathbf{r}_v$ and $\mathbf{p}_v$ values. It also presents the interesting property that $\forall v \in \mathcal{V}$, $\mathbb{P}(X^{(0)} = v) = \mathbf{p}_v$ over the randomness of the secret vector, when distributed uniformly over $[0,1]^{|\mathcal{V}|}$ (demonstration in App. A). In other words, this watermarking does not bias the distribution on expectation over the secret vector.

The detection computes the following score for T tokens:

$$S_T = -\sum_{t=1}^{T} \ln\left(1 - \mathbf{r}_{x^{(t)}}^{(t)} \right). \quad (3)$$

[1](Nucleus sampling can be applied before generating $\mathbf{p}$)

C. Quality-Robustness Trade-off

For both methods we can trade off generation quality against robustness by varying the watermarking strength. In [17], increasing the δ parameter increases the generation of green tokens at the risk of including unlikely tokens. In [16], increasing the temperature θ has the same effect, since it flattens the probability vector (1), thus diminishing the relative importance of $\mathbf{p}_v$ over $\mathbf{r}_v$.

D. Key Management

The secret key k giving birth to the greenlist $\mathcal{G}_k$ in [17] or to the sampling of $\mathbf{r}$ in [16] must have a wide diversity. A fixed key causes security issues and biases the text generation. One possibility is to make it dependent of the time t as proposed in [19]. The secret key is then different from one token to another. Yet, this brings synchronization issue at the detection stage (*e.g.* when a sentence is deleted). A common practice ensuring self-synchronization - illustrated in Fig. 1 - makes the key dependent of the window of h previous tokens: $k^{(t)} = H(x^{(t-1)}, \dots, x^{(t-h)}, k)$, where H is a cryptographic hash function and k the master key. This secret is the seed that initializes a random number generator (RNG) at time t. In turn, the RNG is used to generate the greenlist $\mathcal{G}_{k^{(t)}}$ or to sample $\mathbf{r}^{(t)}$. The width of this window defines a trade-off between diversity of the key and robustness of the watermarking. In the specific case where $h = 0$, the key is the same for all tokens ($k^{(t)} = k$), which makes the watermarking particularly robust to text editing [25].

E. Z-Test

The detection tests the hypothesis $\mathcal{H}_0$: "the text is natural" (human written or written without watermark), against $\mathcal{H}_1$: "the text has been generated with watermark".

Current approaches [16], [17] approximate the underlying distribution of the score S_T by using a Z-test. This statistical hypothesis test determines whether a sample mean differs significantly from its expectation when the standard deviation is known. It computes the so-called Z statistics:

$$Z = \frac{S_T/T - \mu_0}{\sigma_0/\sqrt{T}}, \quad (4)$$

where μ_0 and σ_0 are the expectation and standard deviation per token under the null hypothesis $\mathcal{H}_0$, *i.e.* when the analyzed text is not watermarked. The Z-test is typically used for large sample sizes assuming a normal distribution under the null hypothesis thanks to the central limit theorem. This assumption is key for computing the p-value, *i.e.* the probability of observing a value of Z at least as extreme as the one observed z, under the null hypothesis:

$$\text{p-value}(z) = \mathbb{P}(Z > z | \mathcal{H}_0) = 1 - \Phi(z), \quad (5)$$

where Φ is the cumulative distribution function of the normal distribution. At detection time, we fix a false positive rate (FPR) and flag the text as watermarked if p-value$(z) <$ FPR.

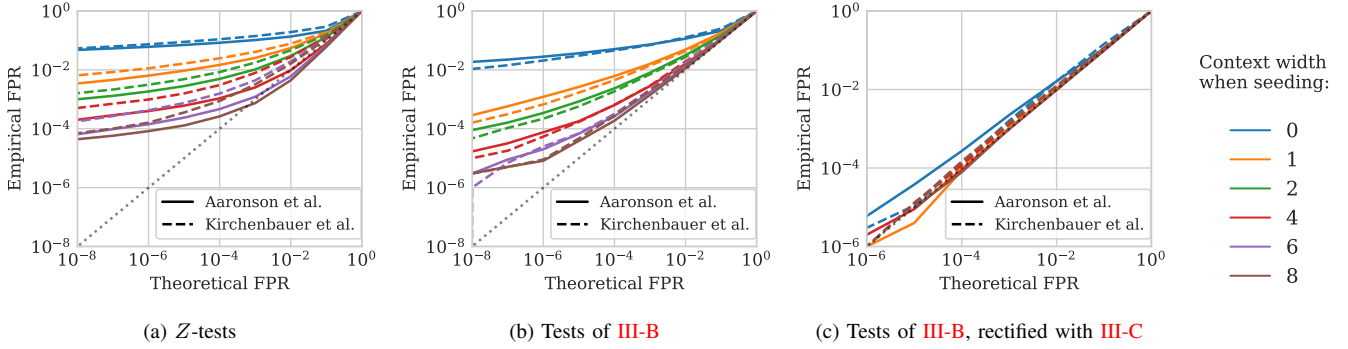| (a) Z-tests | (b) Tests of III-B | (c) Tests of III-B, rectified with III-C |

Fig. 2: Empirical checks of false positive rates for different watermarks and values of the context width h. Results are computed over 10 master keys $\times$ 100k sequences of 256 tokens sampled from Wikipedia. We compare three detection tests: (*Left*) using Z-tests; (*Middle*) using new statistical tests presented in III-B; (*Right*) using the new statistical tests with the rectified scoring strategy of III-C. Theoretical values do not hold in practice for Z-tests, even for high values of h, and empirical FPRs do not match theoretical ones. This is solved by basing detection on grounded statistical tests and analytic p-values, as well as by revising the scoring strategy.

III. RELIABILITY OF THE DETECTION

In this section, large-scale evaluations of the FPR show a gap between theory and practice. It is closed with new statistical tests and by rectifying the scoring method.

A. Empirical Validation of FPR with Z-Scores

So far, the FPR has been checked on only around 500 negative samples [17], [18], [20]. We scale this further and select 100k texts from multilingual Wikipedia to cover the distribution of natural text. We tokenize with LLaMA's tokenizer, and take $T = 256$ tokens/text. We run detection tests with varying window length h when seeding the RNG. We repeat this with 10 different master keys, which makes 1M detection results under $\mathcal{H}_0$ for each method and h value. For the detection of the greenlist watermark, we use $\gamma = 0.25$.

Fig. 2a compares empirical and theoretical FPRs. Theoretical guarantees do not hold in practice: the empirical FPRs are much higher than the theoretical ones. We also observed that distributions of p-values were not uniform (which should be the case under $\mathcal{H}_0$). Besides, the larger the watermarking context window h, the closer we get to theoretical guarantees. In pratice, one would need $h >> 8$ to get reliable p-values, but this makes the watermarking method less robust to attacks on generated text because it hurts synchronization.

B. New Non-Asymptotical Statistical Tests

The Gaussian assumption of Z-tests breaks down for short or repetitive texts. Here are non-asymptotical tests for both methods that reduce the gap between empirical and theoretical FPR, especially at low FPR values as shown in Fig. 2.

1) Kirchenbauer et al. [17]: Under $\mathcal{H}_0$, we assume that the event $x^{(t)} \in \mathcal{G}_{k^{(t)}}$ occurs with probability γ, and that these events are i.i.d. Therefore, S_T (2) is distributed as a binomial of parameters T and γ. Consider a text under scrutiny whose score equals s. The p-value is defined as the probability of obtaining a score higher than s under $\mathcal{H}_0$:

$$\text{p-value}(s) = \mathbb{P}(S_T > s | \mathcal{H}_0) = I_\gamma(s, T - s + 1), \quad (6)$$

because $S \sim \mathcal{B}(T, \gamma)$ whose c.d.f. is expressed by $I_x(a, b)$ the regularized incomplete Beta function.

2) Aaronson et al. [16]: Under $\mathcal{H}_0$, we assume that the text under scrutiny and the secret vector are independent, so that $\mathbf{r}_{x^{(t)}} \overset{i.i.d.}{\sim} \mathcal{U}(0, 1)$. Therefore, S_T (3) follows a $\Gamma(T, 1)$ distribution. The p-value associated to a score s reads:

$$\text{p-value}(s) = \mathbb{P}(S_T > s | \mathcal{H}_0) = \frac{\Gamma(T, s)}{\Gamma(T)}, \quad (7)$$

where Γ is the upper incomplete gamma function. Under $\mathcal{H}_1$, the score is expected to be higher as proven in App. A, so the p-value is likely to be small.

C. Rectifying the Detection Scores

Even with grounded statistical tests, empirical FPRs are still higher than theoretical ones. In fact, Kirchenbauer et al. [17] mention that random variables are only pseudo-random since repeated windows generate the same secret. This can happen even in a short text and especially in formatted data. For instance in a bullet list, the sequence of tokens \n\n*_ repeats a lot as shown in Fig. 3. Repetition pulls down the assumption of independence necessary for computing the p-values.

We experimented with two simple heuristics mitigating this issue at the detection stage. The first one takes into account a token only if the watermark context window has not already been seen during the detection. The second scores the tokens for which the $h + 1$-tuple formed by {watermark context + current token} has not already been seen. Note, that the latter is present in [17], although without ablation and without being

Fig. 3: Typical example of a vanilla text with low p-value because of repeated tokens. It is 10^{-21}, using the greenlist watermark with $\gamma = 1/4$ and $h = 2$ on 256 tokens (we only show half of the text).

TABLE I: Performances on classical free-form generation benchmarks when completion is done with watermarking. h is the watermark context width. We report results for methods: Aaronson *et al.* [16] / Kirchenbauer *et al.* [17]. "-" means no watermarking.

Model	h	GSM8K	Human Eval	MathQA	MBPP	NQ	TQA	Average
7B	-	10.3	12.8	3.0	18.0	21.7	56.9	20.5
	1	10.3 / 11.1	12.8 / 9.8	2.9 / 2.8	18.2 / 16.0	21.8 / 19.5	56.9 / 55.3	20.5 / 19.1
	4	10.4 / 10.8	12.8 / 9.2	3.0 / 2.8	17.8 / 16.4	21.8 / 20.2	56.9 / 55.1	20.4 / 19.1
13B	-	17.2	15.2	4.3	23.0	28.2	63.6	25.3
	1	17.2 / 17.3	15.2 / 14.6	4.3 / 3.6	22.8 / 21.2	28.2 / 25.1	63.6 / 62.2	25.2 / 24.0
	4	17.2 / 16.8	15.2 / 15.9	4.2 / 4.1	22.6 / 21.2	28.2 / 24.5	63.6 / 62.2	25.2 / 24.1
30B	-	35.1	20.1	6.8	29.8	33.5	70.0	32.6
	1	35.3 / 35.6	20.7 / 20.7	6.9 / 7.5	29.6 / 28.8	33.5 / 31.6	70.0 / 69.0	32.7 / 32.2
	4	35.1 / 34.1	20.1 / 22.6	6.9 / 7.0	29.8 / 28.8	33.5 / 31.6	70.0 / 68.7	32.6 / 32.1

used in further experiments. Of the two, the second one is better since it counts more ngrams, and thus has better TPR. It can also deal with the specific case of $h = 0$.

Figure 2c reports empirical and theoretical FPRs when choosing not to score already seen $h + 1$-tuples. They now match perfectly, except for $h = 0$ where the FPR is still slightly underestimated. *In short, we guarantee FPR thanks to new statistical tests and by scoring only tokens for which {watermark context + current token} has not been scored.*

IV. WATERMARK EVALUATION

This section introduces evaluation with the revised statistical tests, and investigate the impact of LLM watermarking on classical NLP benchmarks.

A. Robustness Analysis

We now compare watermarking methods by analyzing the TPR when detecting watermarked texts. For detection, we employ the previous statistical tests and scoring strategy. We flag a text as watermarked if its p-value is lower than 10^{-5} ensuring an FPR=10^{-5}. For these experiments, we stay close to a chatbot scenario. We prompt Guanaco-7b [26], an instruction fine-tuned version of LLaMA, with the first 1k prompts from the Alpaca dataset [27]. For generation, we use top-p sampling with $p = 0.95$, and in the case of [17] a temperature $\theta = 0.8$ and $\gamma = 1/4$. We simulate synonym attacks by randomly replacing tokens with probability 0.3 (other attacks are studied in related work [18]).

Tab. II reports the TPR for different strength of the watermark (see Sect. II-C), and the S-BERT [28] similarity score between the generated texts with and without watermarking to measure the semantic distortion induced by the watermark. Results in Tab. II reveals different behaviors. For instance, [17] has a finer control over the trade-off between watermark strength and quality. Its TPR values ranges from 0.0 to 0.9, while [16] is more consistent but fails to achieve TPR higher than 0.8 even when the S-BERT score is degraded a lot.

The watermark context width also has a big influence. When h is low, we observed that repetitions happen more often because the generation is easily biased towards certain repetitions of tokens. It leads to average S-BERT scores below 0.5 and unusable completions. On the other hand, low h also makes the watermark more robust, especially for [17]. It is also important to note that h has an influence on the number

TABLE II: Robustness analysis of the watermarks, with rectified statistical tests. We report the TPR@FPR=10^{-5} and the S-BERT scores over $10 \times 1k$ completions, for different hyperparameters controling the strength of the watermark (δ in [17] and θ in [16] - see Sect. II-C). The 'TPR aug.' is the TPR when texts are attacked before detection by randomly replacing tokens with probability 0.3.

h	Metric	Aaronson *et al.* [16]				Kirchenbauer *et al.* [17]			
		θ : 0.8	0.9	1.0	1.1	δ : 1.0	2.0	3.0	4.0
0	S-BERT	0.60	0.56	0.52	0.44	0.63	0.61	0.57	0.50
	TPR	0.20	0.31	0.42	0.51	0.00	0.16	0.58	0.70
	TPR aug.	0.04	0.06	0.09	0.10	0.00	0.02	0.20	0.39
1	S-BERT	0.62	0.61	0.59	0.55	0.63	0.62	0.60	0.56
	TPR	0.35	0.51	0.66	0.77	0.02	0.41	0.77	0.88
	TPR aug.	0.04	0.10	0.20	0.36	0.00	0.05	0.30	0.58
4	S-BERT	0.62	0.62	0.61	0.59	0.62	0.62	0.60	0.57
	TPR	0.43	0.59	0.71	0.80	0.02	0.44	0.76	0.88
	TPR aug.	0.01	0.02	0.06	0.18	0.00	0.00	0.03	0.14

of analyzed tokens since we only score tokens for which the $h + 1$-tuple has not been seen before (see Sect. III-C). If h is high, almost all these tuples are new, while if h is low, the chance of repeated tuples increases. For instance in our case, the average number of scored tokens is around 100 for $h = 0$, and 150 for $h = 1$ and $h = 4$.

B. Impact of Watermarks on Free-Form Generation Tasks

Previous studies measure the impact on quality using distortion metrics such as perplexity or similarity score as done in Tab. II. However, such metrics are not informative of the utility of the model for downstream tasks [24], where the real interest of LLMs lies. Indeed, watermarking LLMs could be harmful for tasks that require very precise answers. This section rather quantifies the impact on typical NLP benchmarks, in order to assess the practicality of watermarking.

LLMs are typically evaluated either by comparing samples of plain generation to a set of target references (free-form generation) or by comparing the likelihood of a predefined set of options in a multiple choice question fashion. The latter makes little sense in the case of watermarking, which only affects sampling. We therefore limit our evaluations to free-form generation tasks. We use the evaluation setup of LLaMA: 1) Closed-book Question Answering (Natural Questions [29], TriviaQA [30]): we report the 5-shot exact match performance; 2) Mathematical reasoning (MathQA [31], GSM8k [32]), we report exact match performance without majority voting; 3) Code generation (HumanEval [33], MBPP [34]), we report the pass@1 scores. For [17], we shift logits with $\delta = 1.0$

before greedy decoding. For [16], we apply top-p at 0.95 to the probability vector, then apply the watermarked sampling.

Tab. I reports the performance of LLaMA models on the aforementioned benchmarks, with and without the watermark and for different window size h. The performance of the LLM is not significantly affected by watermarking. The approach of Kirchenbauer *et al.* (II-B1) is slightly more harmful than the one of Aaronson *et al.* (II-B2), but the difference w.r.t. the vanilla model is small. Interestingly, this difference decreases as the size of the model increases: models with higher generation capabilities are less affected by watermarking. A possible explanation is that the global distribution of the larger models is better and thus more robust to small perturbations. Overall, evaluating on downstream tasks points out that watermarking may introduce factual errors that are not well captured by perplexity or similarity scores.

V. ADVANCED DETECTION SCHEMES

This section introduces improvements to the detection schemes of Sect. III. Namely, it develops a statistical test when access to the LLM is granted, as well as multi-bit decoding.

A. Neyman-Pearson and Simplified Score Function

The following is specific for the scheme of Aaronson *et al.* [16] (a similar work may be conducted with [18]). Under $\mathcal{H}_0$, we have $\mathbf{r}_v \sim \mathcal{U}_{[0,1]}$, whereas $\mathbf{r}_v \sim Beta(1/p_v, 1)$ under $\mathcal{H}_1$ (see Corollary (14) in App. A). The optimal Neyman-Pearson score function is thus:

$$S_T = \sum_{t=1}^{T} \ln \frac{f_{\mathcal{H}_1}(\mathbf{r}_{x^{(t)}})}{f_{\mathcal{H}_0}(\mathbf{r}_{x^{(t)}})} = \sum_{t=1}^{T} \left(\frac{1}{\mathbf{p}_{x^{(t)}}} - 1 \right) \ln(\mathbf{r}_{x^{(t)}}) + A$$

where A doesn't depend on $\mathbf{r}$ and can thus be discarded. There are two drawbacks: (1) detection needs the LLM to compute $\mathbf{p}_{x^{(t)}}$, (2) there is no close-form formula for the p-value.

This last point may be fixed by resorting to a Chernoff bound, yet without guarantee on its tightness: p-value$(s) \leq e^{\sum_t \ln \frac{\lambda_t}{\lambda_t + c} - cs}$, with c solution of $\sum_t (c + \lambda_t)^{-1} = -s$ and $\lambda_t = p_{x^{(t)}}/(1 - p_{x^{(t)}})$. Experiments show that this detection yields extremely low p-values for watermarked text, but they are fragile: any attack increases them to the level of the original detection scheme (3), or even higher because generated logits are sensitive to the overall LLM context. An alternative is to remove weighting:

$$S_T = \sum_{t=1}^{T} \ln \left(\mathbf{r}_{x^{(t)}} \right), \qquad (8)$$

whose p-value is given by: p-value$(s) = \frac{\gamma(T, -s)}{\Gamma(T)}$. In our experiments, this score function does not match the original detection presented in [16].

B. Multi-bit Watermarking

1) Theory: It is rather easy to turn a zero-bit watermarking scheme into multi-bit watermarking, by associating a secret key per message. The decoding runs detection with every key and the decoded message is the one associated to the key giving the lowest p-value p. The global p-value becomes $1 - (1 - p)^M$, where M is the number of possible messages.

Algorithm 1 Multi-bit watermarking for LLMs

Requires: model LLM, secret's dimension $d = \max(M, |\mathcal{V}|)$, watermark context width h, message $m \in \{0, \ldots, M-1\}$

Generation (one step):
  logits $\boldsymbol{\ell} \leftarrow \text{LLM}\left(x^{(-C)}, \ldots, x^{(-1)}\right)$
  seed $\leftarrow \text{Hash}(x^{(-h)}, \ldots, x^{(-1)})$
  $\mathbf{r} \leftarrow \text{RNG}_{\text{seed}}(d)$
  $\mathbf{r}(m) \leftarrow \text{CyclicShift}(\mathbf{r}, m) = (\mathbf{r}_m, .., \mathbf{r}_d, \mathbf{r}_0, .., \mathbf{r}_{m-1})$
  $x^{(0)} \leftarrow \text{Sample}(\boldsymbol{\ell}, \mathbf{r}(m)_{1, \ldots, |\mathcal{V}|})$

Identification:
  $\mathbf{S} \leftarrow \mathbf{0}_d$
  for $t \in \{h, \ldots, T\}$:
    seed $\leftarrow \text{Hash}(x^{(t-h)}, \ldots, x^{(t-1)})$
    $\mathbf{r}^{(t)} \leftarrow \text{RNG}_{\text{seed}}(d)$
    $\mathbf{S} \leftarrow \mathbf{S} + \text{CyclicShift}(f(\mathbf{r}^{(t)}), x^{(t)})$
  $\mathbf{p} \leftarrow \text{p-value}(\mathbf{S}_{1, \ldots, M})$
  $m \leftarrow \text{argmin}(\mathbf{p})$
  $p \leftarrow 1 - (1 - \mathbf{p}_m)^M$

Running detection for M keys is costly, since it requires M generations of the secret vector. This is solved by imposing that the secret vectors of the messages $m \in \{0, \ldots, M-1\}$ are crafted as circular shifts of m indices of $\mathbf{r} = \mathbf{r}(0)$:

$$\mathbf{r}(m) = \text{CyclicShift}(\mathbf{r}, m)$$
$$= (\mathbf{r}_m, \mathbf{r}_{m+1}, .., \mathbf{r}_d, \mathbf{r}_0, .., \mathbf{r}_{m-1}).$$

Generating $\mathbf{r}$ as a d-dimensional vector, with $d \geq |\mathcal{V}|$, we are able to embed $M \leq d$ different messages, by keeping only the first $|\mathcal{V}|$ dimensions of each circularly-shifted vector. Thus, the number of messages may exceed the size of the token vocabulary $|\mathcal{V}|$.

Besides, the scoring functions (2) (3) may be rewritten as:

$$S_T(m) = \sum_{t=1}^{T} f\left(\mathbf{r}^{(t)}(m)\right)_{x^{(t)}}, \qquad (9)$$

where $f : \mathbb{R}^d \mapsto \mathbb{R}^d$ is a component-wise function, and $x^{(t)}$ is the selected token during detection. This represents the selection of $f\left(\mathbf{r}^{(t)}(m)\right)$ at position $x^{(t)}$. From another point of view, if we shift $f\left(\mathbf{r}^{(t)}\right)$ by $x^{(t)}$, the score for $m = 0$ would be its first component, $m = 1$ its second one, etc. We may also write:

$$\mathbf{S}_T = \sum_{t=1}^{T} \text{CyclicShift}\left(f\left(\mathbf{r}^{(t)}\right), x^{(t)}\right), \qquad (10)$$

and the first M components of $\mathbf{S}_T$ are the scores for each m. As a side note, this is a particular case of the parallel computations introduced by Kalker *et al.* [35].

2) Experiments: In a tracing scenario the message is the identifier of a user or a version of the model. The goal is to decide if any user or model generated a given text (detection) and if so, which one (identification). There are 3 types of error: *false positive*: flag a vanilla text; *false negative*: miss a watermarked text; *false accusation*: flag a watermarked text but select the wrong identifier.

We simulate M'=1000 users that generate 100 watermarked texts each, using the Guanaco-7b model. Accuracy can then be extrapolated beyond the M' identifiers by adding identifiers

TABLE III: Identification accuracy for tracing users by watermarking. Sequences are between 4 and 252 tokens long, and 149 on average.

	Number of users M	10	10^2	10^3	10^4
FPR$= 10^{-3}$	Aaronson *et al*. [16]	0.80	0.72	0.67	0.62
	Kirchenbauer *et al*. [17]	0.84	0.77	0.73	0.68
FPR$= 10^{-6}$	Aaronson *et al*. [16]	0.61	0.56	0.51	0.46
	Kirchenbauer *et al*. [17]	0.69	0.64	0.59	0.55

with no associated text, for a total of $M > M'$ users. Text generation uses nucleus sampling with top-p at 0.95. For [17], we use $\delta = 3.0$, $\gamma = 1/4$ with temperature θ at 0.8. For [16], we use $\theta = 1.0$. For both, the context width is $h = 4$. A text is deemed watermarked if the score is above a threshold set for a given *global* FPR (see III). Then, the source is identified as the user with the lowest p-value.

Tab. III shows that watermarking enables identification because its performance is dissuasive enough. For example, among 10^5 users, we successfully identify the source of a watermarked text 50% of the time while maintaining an FPR of 10^{-6} (as long as the text is not attacked). At this scale, the false accusation rate is zero (no wrong identification once we flag a generated text) because the threshold is set high to avoid FPs, making false accusations unlikely. The identification accuracy decreases when M increases, because the threshold required to avoid FPs gets higher. In a nutshell, by giving the possibility to encode several messages, we trade some accuracy of detection against the ability to identify users.

VI. CONCLUSION

This research offers theoretical and empirical insights that were kept aside from the literature on watermarks for LLMs. Namely, existing methods resort to statistical tests which are biased, delivering incorrect false positive rates. This is fixed with grounded statistical tests and a revised scoring strategy. We additionally introduce evaluation setups, and detection schemes to consolidate watermarks for LLMs. Further work may investigate how to adapt watermarks for more complex sampling schemes (*e.g.* beam search as in [17]), since generation yield significantly better quality with these methods.

Overall, we believe that watermarking is both reliable and practical. It already holds many promises as a technique for identifying and tracing LLM outputs, while being relatively new in the context of generative models.

REFERENCES

[1] OpenAI, "ChatGPT: Optimizing language models for dialogue.," 2022.
[2] AnthropicAI, "Introducing Claude," 2023.
[3] H. Touvron *et al.*, "Llama: Open and efficient foundation language models," *arXiv*, 2023.
[4] L. Weidinger *et al.*, "Taxonomy of risks posed by language models," in *ACM Conference on Fairness, Accountability, and Transparency*, 2022.
[5] E. Crothers, N. Japkowicz, and H. Viktor, "Machine generated text: A comprehensive survey of threat models and detection methods," *arXiv*, 2022.
[6] J. P. Cardenuto, J. Yang, R. Padilha, R. Wan, D. Moreira, H. Li, S. Wang, F. Andaló, S. Marcel, and A. Rocha, "The age of synthetic realities: Challenges and opportunities," *arXiv*, 2023.
[7] K. Kertysova, "Artificial intelligence and disinformation: How AI changes the way disinformation is produced, disseminated, and can be countered," *Security and Human Rights*, vol. 29, no. 1-4, 2018.
[8] S. Kreps, R. M. McCain, and M. Brundage, "All the news that's fit to fabricate: Ai-generated text as a tool of media misinformation," *Journal of experimental political science*, 2022.
[9] E. Kušen and M. Strembeck, "Politics, sentiments, and misinformation: An analysis of the twitter discussion on the 2016 austrian presidential elections," *Online Social Networks and Media*, 2018.
[10] M. T. review, "Junk websites filled with ai-generated text are pulling in money from programmatic ads," 2023.
[11] D. Ippolito, D. Duckworth, C. Callison-Burch, and D. Eck, "Automatic detection of generated text is easiest when humans are fooled," *arXiv*, 2019.
[12] E. Mitchell, Y. Lee, A. Khazatsky, C. D. Manning, and C. Finn, "Detectgpt: Zero-shot machine-generated text detection using probability curvature," *arXiv*, 2023.
[13] N. Yu, V. Skripniuk, D. Chen, L. Davis, and M. Fritz, "Responsible disclosure of generative models using scalable fingerprinting," in *ICLR*, 2022.
[14] P. Fernandez, G. Couairon, H. Jégou, M. Douze, and T. Furon, "The stable signature: Rooting watermarks in latent diffusion models," *ICCV*, 2023.
[15] Y. Wen, J. Kirchenbauer, J. Geiping, and T. Goldstein, "Tree-ring watermarks: Fingerprints for diffusion images that are invisible and robust," *arXiv*, 2023.
[16] S. Aaronson and H. Kirchner, "Watermarking GPT Outputs," 2023.
[17] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, and T. Goldstein, "A watermark for large language models," *ICML*, 2023.
[18] J. Kirchenbauer, J. Geiping, Y. Wen, M. Shu, K. Saifullah, K. Kong, K. Fernando, A. Saha, M. Goldblum, and T. Goldstein, "On the reliability of watermarks for large language models," 2023.
[19] M. Christ, S. Gunn, and O. Zamir, "Undetectable watermarks for language models," *Cryptology ePrint Archive*, 2023.
[20] X. Zhao, P. Ananth, L. Li, and Y.-X. Wang, "Provable robust watermarking for ai-generated text," *arXiv*, 2023.
[21] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," *NeurIPS*, vol. 13, 2000.
[22] A. Fan, M. Lewis, and Y. Dauphin, "Hierarchical neural story generation," *arXiv*, 2018.
[23] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, *et al.*, "Language models are unsupervised multitask learners," *OpenAI*, 2019.
[24] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," *arXiv*, 2019.
[25] X. Zhao, Y.-X. Wang, and L. Li, "Protecting language generation models via invisible watermarking," *arXiv*, 2023.
[26] T. Dettmers, A. Pagnoni, A. Holtzman, and L. Zettlemoyer, "Qlora: Efficient finetuning of quantized llms," *arXiv*, 2023.
[27] R. Taori, I. Gulrajani, T. Zhang, Y. Dubois, X. Li, C. Guestrin, P. Liang, and T. B. Hashimoto, "Stanford Alpaca: An instruction-following LLaMA model," 2023.
[28] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv*, 2019.
[29] T. Kwiatkowski *et al.*, "Natural questions: a benchmark for question answering research," *Trans. of the ACL*, vol. 7, 2019.
[30] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer, "Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension," *arXiv*, 2017.
[31] D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt, "Measuring mathematical problem solving with the math dataset," *arXiv*, 2021.
[32] K. Cobbe *et al.*, "Training verifiers to solve math word problems," *arXiv*, 2021.
[33] M. Chen *et al.*, "Evaluating large language models trained on code," *arXiv*, 2021.
[34] J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le, and C. Sutton, "Program synthesis with large language models," 2021.
[35] T. Kalker, G. Depovere, J. Haitsma, and M. Maes, "Video watermarking system for broadcast monitoring," in *Proc. of SPIE, Security and Watermarking of Multimedia Contents*, vol. 3657, 1999.

A. Demonstrations for [16]

1) Sampling probability

Proposition. Consider a discrete distribution $\mathbf{p} = (p_1, \ldots, p_V)$ and V random variables $\mathbf{R} = (R_1, \ldots, R_V)$ s.t. $R_v \overset{iid}{\sim} \mathcal{U}_{[0,1]}$. Let $V^\star = \arg\max_v R_v^{1/p_v}$. Then $\mathbb{P}(V^\star = v) = p_v$.

Proof. For any $v \in \mathcal{V}$, $R_v \overset{iid}{\sim} \mathcal{U}_{[0,1]}$ so, $-\ln(R_v)$ follows an exponential distribution $\mathcal{E}(1)$. Let $Z_v := -\frac{1}{p_v}\ln(R_v)$. By construction, $Z_v \sim \mathcal{E}(p_v)$, with density $f_{Z_v}(z) = p_v e^{-p_v \cdot z}$. We now have:

$$V^\star = \arg\max_v R_v^{\frac{1}{p_v}} = \arg\min_v Z_v. \tag{11}$$

A well known result about exponential laws is that (see the-gumbel-trick for following lines):

$$\underline{Z} = \min_v Z_v \sim \mathcal{E}\left(\sum_v p_v\right) = \mathcal{E}(1), \tag{12}$$

$$\mathbb{P}(V^\star = v) = \frac{p_v}{\sum_j p_j} = p_v. \tag{13}$$

This shows that for a given secret vector $\mathbf{r}$, the watermarking chooses a word which may be unlikely (low probability $p_{V^\star}$). Yet, on expectation over the secret keys, ie. over r.v. $\mathbf{R} = (R_1, \ldots, R_V)$, the distribution of the chosen token follows the distribution given by the LLM.

Corollary. $R_{V^\star} \sim Beta(1/p_{V^\star}, 1)$.

Proof.

$$\underline{Z} = Z_{V^\star} = -\frac{1}{p_{V^\star}}\ln(R_{V^\star}) \sim \mathcal{E}(1), \tag{14}$$

which translates to $R_{V^\star} = e^{-p_{V^\star}E}$ with $E \sim \mathcal{E}(1)$, with p.d.f. $f_{R_{V^\star}}(r) = \frac{r^{\frac{1}{p_{V^\star}}-1}}{p_{V^\star}}$. Therefore, $R_{V^\star} \sim Beta(1/p_{V^\star}, 1)$.

2) Detection

We denote by $x^{(1)}, \ldots, x^{(T)}$ the sequence of tokens in the text, by $\mathbf{p}^{(t)}$ the probability vector output by the LLM and by $\mathbf{R}^{(t)} \in [0,1]^{|\mathcal{V}|}$ the key random vector at time-step t. We define $R_t := R_{x^{(t)}}^{(t)}$ and $p_t := p_{x^{(t)}}^{(t)}$ at time-step t. The score is $S_T = -\sum_{t=1}^{T}\ln(1 - R_t)$.

Proposition (p-value under $\mathcal{H}_0$). The p-value associated to a score s is defined as:

$$p\text{-value}(s) = \mathbb{P}(S_T > s|\mathcal{H}_0) = \frac{\Gamma(T, s)}{\Gamma(T)}, \tag{15}$$

where $\Gamma(T, s)$ is the *upper* incomplete gamma function.

Proof. Under $\mathcal{H}_0$, the assumption is s.t. $R_t \overset{iid}{\sim} \mathcal{U}_{[0,1]}$. Then, $-\ln(1 - R_t)$ follows an exponential distribution $\mathcal{E}(1)$. Therefore $S \sim \Gamma(T, 1)$ (see sum of Gamma distributions). Therefore the p-value associated to a score s is

$$p\text{-value}(s) = 1 - \frac{\gamma(T, s)}{\Gamma(T)} = \frac{\Gamma(T, s)}{\Gamma(T)}, \tag{16}$$

where $\Gamma(T, s)$ is the *upper* incomplete gamma function, $\gamma(T, s)$ is the *lower* incomplete gamma function.

Corollary. Per token,

$$\mu_0 = \mathbb{E}(S_T/T|\mathcal{H}_0) = 1, \quad \sigma_0^2 = \mathbb{V}(S_T/T|\mathcal{H}_0) = 1/T. \tag{17}$$

Proposition (Bound on expected score under $\mathcal{H}_1$). Under $\mathcal{H}_1$, $\mathbb{E}(S_T) \geq T + \left(\frac{\pi^2}{6} - 1\right)H_T$, where $H_T = -\sum_{t=1}^{T} p_t \ln(p_t)$ is the entropy of the completion.

Proof. From (14), $R_t = \exp(-p_t E)$ with $E \sim \mathcal{E}(1)$, so:

$$\mathbb{E}(S) = -\mathbb{E}\left[\sum_{t=1}^{T} \ln(1 - \exp(-p_t E))\right]$$

$$= -\sum_{t=1}^{T} \int_0^{\infty} \ln(1 - e^{-p_t x}) e^{-x} dx$$

$$= -\sum_{t=1}^{T} \int_0^1 \frac{1}{p_t} r^{1/p_t - 1}(-\ln(1-r)) dr$$

(by change of variable $x = -1/p_t \ln(r)$)

Then, using integration by parts with $u = 1 - r^{1/p_t}$ and $v = \ln(1-r)$, the integral becomes:

$$-\int_0^1 \frac{1}{p_t} r^{1/p_t - 1} \ln(1-r) dr = \int_0^1 \frac{1 - r^{1/p_t}}{1 - r} dr = \mathcal{H}_{1/p_t}$$

where $\mathcal{H}_z$ is the z-th harmonic number also defined as $\mathcal{H}_z = \sum_{n=1}^{\infty} \frac{1}{n} - \frac{1}{n+z}$. Therefore, we have:

$$-\int_0^1 \frac{1}{p_t} r^{1/p_t - 1} \ln(1-r) dr = \sum_{n=1}^{\infty} \frac{1}{n} - \frac{1}{n + 1/p_t}$$

$$= 1 + \sum_{n=1}^{\infty} \frac{1}{n+1} - \frac{1}{n + 1/p_t}.$$

Now, $\forall n \in \mathbb{N}^\star$, we have:

$$(n+1)^2 \left(\frac{1}{n+1} - \frac{1}{n + 1/p_t}\right) = \frac{(n+1)(n + 1/p_t) - (n+1)^2}{n + 1/p_t}$$

$$= \frac{1+n}{1/p_t + n}(1/p_t - 1)$$

$$\geq -\frac{1+n}{1/p_t + n} \ln(p_t)$$

$$\geq -p_t \ln(p_t).$$

Therefore, by summing over all $t \in [1, T]$,

$$\mathbb{E}(S) \geq T + \left(\sum_{n=1}^{\infty} \frac{1}{(n+1)^2}\right)\left(\sum_{t=1}^{T} -p_t \ln(p_t)\right)$$

$$= T + \left(\frac{\pi^2}{6} - 1\right) H_T.$$

Proposition (Variance of score under $\mathcal{H}_1$). $\mathbb{V}(S_T) \leq T\frac{\pi^2}{6}$.

Proof. For $R_t \sim Beta(1/p_t, 1)$:

$$\mathbb{V}(\ln(1 - R_t)) = \psi_1(1) - \psi_1(1 + 1/p_t) \tag{18}$$

where ψ_1 is the trigamma function, which can be expressed as the following serie $\psi_1(z) = \sum_{n=0}^{\infty} 1/(n + z)^2$. Then $\psi_1(1) = \pi^2/6$ and $\psi_1(1 + 1/p_t) > 0$, so that $\mathbb{V}(\ln(1 - R_t)) \leq \pi^2/6$. The results comes because the sampled tokens are independent.

B. Free-form evaluations

We provide in Table IV the full results of the free-form evaluations of the different models. This extends the results of Table I in the main paper. The models are evaluated with the same evaluation protocol as in LLaMA.

TABLE IV

Model	WM Method	h	GSM8K	Human Eval	MathQA	MBPP	NQ	TQA	Average
7B	None	-	10.31	12.80	2.96	18.00	21.72	56.89	20.45
	Aaronson *et al.*	0	10.54	12.80	3.00	18.00	21.77	56.88	20.50
		1	10.31	12.80	2.88	18.20	21.75	56.87	20.47
		2	10.31	12.80	2.94	18.00	21.75	56.86	20.44
		3	10.39	12.80	2.96	18.20	21.69	56.85	20.48
		4	10.39	12.80	2.98	17.80	21.80	56.88	20.44
		6	10.61	12.80	2.96	18.00	21.75	56.86	20.50
		8	10.46	12.80	2.90	18.20	21.75	56.85	20.49
	Kirchenbauer *et al.*	0	9.63	12.80	2.20	16.20	20.06	55.09	19.33
		1	11.14	9.76	2.82	16.00	19.50	55.30	19.09
		2	11.07	6.71	2.62	16.00	20.44	55.07	18.65
		3	10.16	10.98	2.38	14.40	20.08	55.65	18.94
		4	10.77	9.15	2.76	16.40	20.17	55.14	19.06
		6	10.01	9.76	3.16	17.00	20.78	54.90	19.27
		8	11.37	11.59	2.90	16.40	20.66	55.36	19.71
13B	None	-	17.21	15.24	4.30	23.00	28.17	63.60	25.25
	Aaronson *et al.*	0	17.29	15.24	4.24	22.80	28.17	63.60	25.22
		1	17.21	15.24	4.30	22.80	28.20	63.61	25.23
		2	17.51	15.24	4.20	22.80	28.20	63.59	25.26
		3	17.44	15.24	4.22	22.60	28.23	63.57	25.22
		4	17.21	15.24	4.20	22.60	28.20	63.63	25.18
		6	16.98	15.24	4.28	23.20	28.23	63.61	25.26
		8	17.21	15.24	4.22	22.80	28.20	63.62	25.22
	Kirchenbauer *et al.*	0	14.33	14.02	3.04	20.80	24.32	62.13	23.11
		1	17.29	14.63	3.62	21.20	25.12	62.23	24.02
		2	16.45	11.59	3.54	20.60	25.54	62.44	23.36
		3	17.06	16.46	3.58	19.80	25.90	62.37	24.20
		4	16.76	15.85	4.08	21.20	24.49	62.24	24.10
		6	15.85	14.63	4.00	18.20	26.32	62.19	23.53
		8	17.29	14.63	3.68	21.00	25.46	62.17	24.04
30B	None	-	35.10	20.12	6.80	29.80	33.55	70.00	32.56
	Aaronson *et al.*	0	35.48	20.12	6.88	29.80	33.52	69.98	32.63
		1	35.33	20.73	6.88	29.60	33.52	70.03	32.68
		2	35.33	20.73	6.94	30.00	33.49	70.00	32.75
		3	35.71	20.73	6.92	30.00	33.52	70.02	32.82
		4	35.10	20.12	6.90	29.80	33.49	70.01	32.57
		6	35.33	20.73	6.86	29.80	33.49	69.98	32.70
		8	35.33	20.73	6.94	30.00	33.52	70.01	32.75
	Kirchenbauer *et al.*	0	31.84	21.95	6.88	28.40	31.66	69.03	31.63
		1	35.56	20.73	7.54	28.80	31.58	68.98	32.20
		2	33.21	17.07	6.48	27.40	31.83	69.44	30.91
		3	33.89	24.39	6.54	27.80	32.49	69.22	32.39
		4	34.12	22.56	6.96	28.80	31.55	68.74	32.12
		6	34.34	24.39	7.32	29.80	31.63	69.08	32.76
		8	34.95	20.12	7.42	27.20	32.08	69.31	31.85