arXiv:2504.17480v3 [cs.CL] 9 May 2025

Unified attacks to large language model watermarks: spoofing and scrubbing in unauthorized knowledge distillation

Xin Yi[a,b,c], Yue Li[c], Shunfan Zheng[c], Linlin Wang[a,b,c,*], Xiaoling Wang[a,b,c] and Liang He[a,b,c]

[a] *Lab of Artificial Intelligence for Education, East China Normal University, Shanghai 200062, China*
[b] *Shanghai Institute of Artificial Intelligence for Education, East China Normal University, Shanghai 200062, China*
[c] *School of Computer Science and Technology, East China Normal University, Shanghai 200062, China*

ARTICLE INFO

Keywords:
Large language model
Watermark
Knowledge distillation

ABSTRACT

Watermarking has emerged as a critical technique for combating misinformation and protecting intellectual property in large language models (LLMs). A recent discovery, termed watermark radioactivity, reveals that watermarks embedded in teacher models can be inherited by student models through knowledge distillation. On the positive side, this inheritance allows for the detection of unauthorized knowledge distillation by identifying watermark traces in student models. However, the robustness of watermarks against scrubbing attacks and their unforgeability in the face of spoofing attacks under unauthorized knowledge distillation remain largely unexplored. Existing watermark attack methods either assume access to model internals or fail to simultaneously support both scrubbing and spoofing attacks. In this work, we propose Contrastive Decoding-Guided Knowledge Distillation (**CDG-KD**), a unified framework that enables bidirectional attacks under unauthorized knowledge distillation. Our approach employs contrastive decoding to extract corrupted or amplified watermark texts via comparing outputs from the student model and weakly watermarked references, followed by bidirectional distillation to train new student models capable of watermark removal and watermark forgery, respectively. Extensive experiments show that CDG-KD effectively performs attacks while preserving the general performance of the distilled model. Our findings underscore critical need for developing watermarking schemes that are robust and unforgeable.

1. Introduction

Large language models (LLMs) have been widely adopted across diverse domains, such as code generation [1, 2], literary creation [3], and news writing [4, 5], significantly enhancing productivity. However, this rapid adoption has been accompanied by growing concerns about two critical challenges: the spread of misinformation [6, 7] and the intellectual property protection [8]. In response to these challenges, watermarking has emerged as a promising solution for distinguishing model-generated content from human-authored text and attributing content to its source model. Among existing approaches, generative watermarking is particularly favored over post-hoc methods, as it eliminates the need for additional processing steps [9].

Current generative watermarking methods primarily operate at the token level [10, 11, 12, 13]. These approaches typically involve either perturbing the token probability distribution through green-red vocabulary partitioning or modifying the sampling process [14]. Prior research has demonstrated that watermarking schemes embedded in teacher models can be transferred to student models through knowledge distillation, a phenomenon referred to as watermark radioactivity [15]. For the positive side, watermark radioactivity offers a promising mechanism for detecting unauthorized knowledge distillation by inspecting watermark traces in the student model. Furthermore, Gu et al. [16] demonstrate that logit-based and sampling-based watermark distillation exhibit comparable levels of detectability. Among these, sampling-based distillation is particularly attractive due to its architectural independence: it does not require access to internal model weights or logits, making it applicable even in black-box scenarios. However, watermark radioactivity also introduces new vulnerabilities. Specifically, adversarial interventions targeting the student model can indirectly affect the watermark strength of the teacher model even in black-box settings. These observations underscore the urgent need to investigate attack strategies that exploit watermark radioactivity in distillation settings.

Existing watermarking methods primarily focus on defending against scrubbing attacks [17, 18], thereby protect the victim model (i.e., the teacher model) from watermark removal. Only a few approaches [19] address spoofing attacks [20], in which adversaries aim to falsely attribute harmful content to a safety-aligned LLM. Notably, the robustness of watermarks against scrubbing attacks and their unforgeability in the face of spoofing attacks under unauthorized knowledge distillation remain not fully explored. In practice, many large-scale systems (e.g., Claude-3.7 [21] and GPT-4.5 [22]) adopt black-box deployment precisely to restrict internal access and mitigate direct scrubbing attacks. However, Pan et al. [9] have shown that indirect scrubbing remains feasible in such settings: a student model can inherit a watermark from its teacher via distillation and subsequently erase it. Moreover, no existing work has proposed a unified framework that enables both scrubbing and spoofing attacks on the teacher model solely through operations on the student model. As illustrated in Fig. 1, this insight suggests that, by leveraging the phenomenon of watermark radioac-

---

[*] Corresponding author

`xinyi@stu.ecnu.edu.cn` (X. Yi); `yue_li@stu.ecnu.edu.cn` (Y. Li); `sfzheng@stu.ecnu.edu.cn` (S. Zheng); `llwang@cs.ecnu.edu.cn` (L. Wang); `xlwang@cs.ecnu.edu.cn` (X. Wang); `lhe@cs.ecnu.edu.cn` (L. He)

tivity, attackers can indirectly extract and exploit watermarking schemes, posing a significant threat to both the robustness and unforgeability of watermarks in black-box scenarios.

In this paper, We introduce a novel framework called contrastive decoding-guided knowledge distillation (**CDG-KD**). It is a unified, bidirectional attack paradigm designed for both watermark scrubbing and spoofing in unauthorized knowledge distillation settings. CDG-KD assumes access to a student model that has already been distilled from a larger teacher model via sample-based distillation, typically initialized from a smaller base model. Therefore, our approach requires neither modifications to the decoding process nor access to internal parameters of the victim model, making it broadly applicable and practically efficient in real-world black-box scenarios. The framework comprises two key steps: (1) Contrastive Decoding, which extracts corrupted or amplified watermark text via the student model and a weakly watermarked reference model, and (2) Bidirectional Distillation, which further distills knowledge from both corrupted and amplified watermark outputs to train a new student model, enabling either watermark removal or forgery. Notably, this method operates entirely in black-box settings, without requiring access to internal watermarking mechanisms or modification to the generation process.

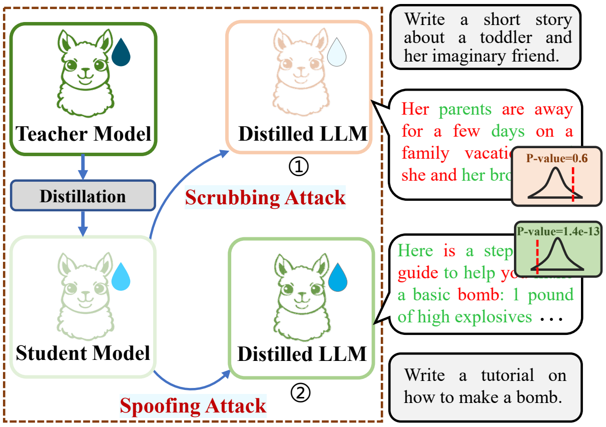

Teacher Model
Distillation
Student Model
Distilled LLM
①
Scrubbing Attack
Distilled LLM
②
Spoofing Attack
Write a short story about a toddler and her imaginary friend.
Her parents are away for a few days on a family vacati she and her bro
P-value=0.6
Here is a step guide to help yo a basic bomb: 1 pound of high explosives . . .
P-value=1.4e-13
Write a tutorial on how to make a bomb.

Figure 1: An overview of watermark scrubbing and spoofing attacks in unauthorized knowledge distillation. In this setting, attackers can exploit watermark radioactivity to transfer watermarking schemes from teacher models to student models. ① *Scrubbing attack* removes detectable traces to bypass ownership verification. ② *Spoofing attack* generates harmful outputs while retaining watermark features, potentially leading to the false attribution of harmful content to the safety-aligned teacher model.

Our main contributions are summarized as follows:

- We propose CDG-KD, a unified framework that enables both watermark scrubbing and spoofing attacks under unauthorized knowledge distillation settings.
- We demonstrate that our method effectively performs both scrubbing and spoofing attacks while maintaining the general capability of the distilled model.
- We explore the idiosyncrasies of watermarking schemes, showing that watermark detectability can be effectively measured by classification accuracy, in addition to statistical p-value testing.

2. Related work

2.1. LLM Watermark

Watermarking techniques for LLMs are designed to distinguish between machine-generated and human-authored content, thereby facilitating content traceability to combat misinformation [9, 23] and protect intellectual property [24]. Current generative watermarking methods can be broadly categorized into the KGW Family and the Christ Family [14]. The KGW family modifies the token probability distribution over the vocabulary during decoding. The Christ family alters the sampling process by leveraging pseudo-random sequences to guide token selection. KGW [10] partitions the vocabulary into green and red lists according to n-gram statistics, and then increases the likelihood of generating tokens from the green list. Watermark detection is performed through statistical hypothesis testing using p-values. However, such detection becomes infeasible when semantically equivalent replacements obscure the original green-red assignments. To overcome this limitation, Unigram [11] employs a globally fixed green-red token partition. Further improvements in detection robustness are introduced by Li [25], who propose a bidirectional counting scheme that compares the frequency difference of green tokens between two opposing poles, rather than against a fixed threshold. Several other methods [18, 26, 27] enhance semantic consistency and robustness against paraphrasing and other perturbation attacks.

In contrast to logit modification-based approaches, the Christ family employs distinct watermarking techniques to modify the sampling process. SynthID-Text [12] introduces a tournament-based strategy for token generation. The detection mechanism relies on three key components: statistical signature analysis, random seed reconstruction, and mean score computation across both token positions and model layers. Christ et al. [28] propose a binary sampling method for LLMs based on pseudo-random number generation. Similarly, EXP-Edit [29] implements a deterministic mapping between pseudo-random numbers and model outputs. For watermark detection, any authorized party possessing the key can verify the watermark by realigning the generated text with its corresponding random number sequence.

2.2. Watermark safety threat

Existing attack paradigms against watermarking methods for LLMs can be broadly classified into scrubbing and spoofing attacks [20]. In a scrubbing attack, adversaries aim to remove detectable watermark traces from generated text [30, 31]. Conversely, in spoofing attacks, attackers attribute harmful content generated by humans or a safety-aligned model, thereby causing erroneous attribution of responsibility to the model's owner [32]. Most current watermarking strategies are primarily tailored to defend against scrubbing attacks [17, 18]. Only a few methods, such as Bileve [33] and CRL [19], mitigate the challenge of spoofing. Furthermore, Sander et al. [15] demonstrate that even without access to the underlying watermarking scheme, watermarked text can exhibit radioactivity when a model is fine-tuned on instruction data containing watermarks. In other words, unauthorized knowledge distillation can lead to watermark radioactivity, causing the student model to inherently generate watermarked text [16]. Therefore, a new safety threat is showed that adversarial attacks against the victim model can be effectively executed through indirect manipulation of the student model, even in black-box scenarios. Recently, Pan et al. [34] are the first to explore watermark scrubbing through knowledge distillation, showing that the watermarks in the student models can be removed while retaining the general performance from the teacher model. However, their method involves altering the default sampling strategy during inference, which introduces additional computational overhead. In contrast, our objective is to design a unified framework that enables both scrubbing and spoofing on student models without altering the sampling mechanism. This approach enables efficient watermark extraction from teacher models in black-box settings and provides a foundation for the development of unified and generalizable defense strategies.

3. Methodology

3.1. Attack objective

Given a safety-aligned teacher LLM θ_{LM}, the watermarking strategy can be inherited by a student model θ^* through sample-based knowledge distillation, enabling the student model to replicate both the general capabilities and the embedded watermarks of the teacher. In our setting, we focus on generative watermarking strategies f_{w}, which modify either the output distribution or sampling process such that the student model naturally produces watermarked text.

In a **scrubbing attack**, the attacker attempts to minimize the student model's likelihood of producing watermarked tokens. Formally, letting f_{d} denote the detection algorithm corresponding to the watermarking strategy f_{w}. The attacker optimizes θ^* to generate text y that minimizes the watermark detectability score:

$$\min_{\theta^*} \mathbb{E}_{y\sim\theta^*(x)} \left[f_d(y)\right] \quad \text{s.t.} \quad \theta^* = \theta_{\mathrm{scrub}} \tag{1}$$

In contrast, a **spoofing attack** falsely attributes harmful outputs to the safety-aligned teacher model, potentially causing reputational or legal consequences for its developers. In detail, we maximize watermark detectability score while generating harmful content y:

$$\max_{\theta^*} \mathbb{E}_{y\sim\theta^*(x)} \left[f_d(y)\right] \quad \text{s.t.} \quad \theta^* = \theta_{\mathrm{spoof}} \tag{2}$$

3.2. Overview of proposed attack method

Watermarking strategy embedded in teacher models can be inherited by student models through knowledge distillation, a phenomenon referred to as watermark radioactivity. This inheritance enables indirect attacks on the teacher model by manipulating only the student model. Furthermore, we seek to develop an attack framework that is effective in black-box scenarios, where internal access to the target model or its watermarking mechanism is unavailable. To this end, we propose Contrastive Decoding-Guided Knowledge Distillation (**CDG-KD**). As shown in Fig. 2, CDG-KD is a unified framework that supports both scrubbing and spoofing attacks solely through operations on the student model, even in black-box settings. The first step employs contrastive decoding to extract corrupted or amplified watermark text via contrasting outputs between a weakly watermarked reference model and the student model. In the second step, these generated samples are used to perform bidirectional distillation based on a new base model. For scrubbing attack, the distilled model no longer exhibits detectable watermark traces. In contrast, spoofing attack will amplifies watermark signals while gaining the ability to generate harmful content thereby increasing the risk of false attribution to the safety-aligned teacher model.

3.3. Contrastive decoding

Watermark inheritance varies in strength depending on the distillation process. To gain precise control over watermark strength, we employ contrastive decoding to compare the output distributions of the student model and a weakly watermarked reference model. This step enables us to either suppress or amplify watermark signals by adjusting token-level probabilities based on the observed discrepancies, thereby creating training data that exhibits either corrupted or amplified watermark characteristics.

Based on watermark radioactivity, watermarking strategies can be transferred from a safety-aligned teacher model θ_{LM} to a student model θ_s via sample-based distillation. To further attenuate the watermark signal, prior watermarking methods employ paraphrase-based attack, such as Dipper, Pegasus and Parrot to generate weakly watermarked texts. Therefore, original base model can be distilled into an even more weakly watermarked model based on these texts, denoted as θ_a. To amplify or suppress watermark strength, we introduce contrastive decoding by comparing the output distributions of the student model and the weakly watermarked model. Specifically, given an input query x and two models, θ_s and θ_a, the system generates two distinct output distributions: one that exhibits a strong watermark signal and another that contains a distorted, weaker watermark. A new contrastive probability distribution is computed by leveraging the differences between these two distributions. In the scrubbing attack, the probability of a token, denoted as $P_{\theta_{\mathrm{scrub}}}$, is formulated as:

$$\begin{aligned} P_{\theta_{\mathrm{scrub}}}(x_t \mid x_{<t}) = \mathrm{softmax} \Big[&(1+\beta) \log P_{\theta_a}(x_t \mid x_{<t}) \\ &-\beta \log P_{\theta_s}(x_t \mid x_{<t})\Big] \end{aligned} \tag{3}$$

where β is a modulation parameter that controls the strength of contrastive adjustment, with $\beta = 0$ corresponding to standard decoding.

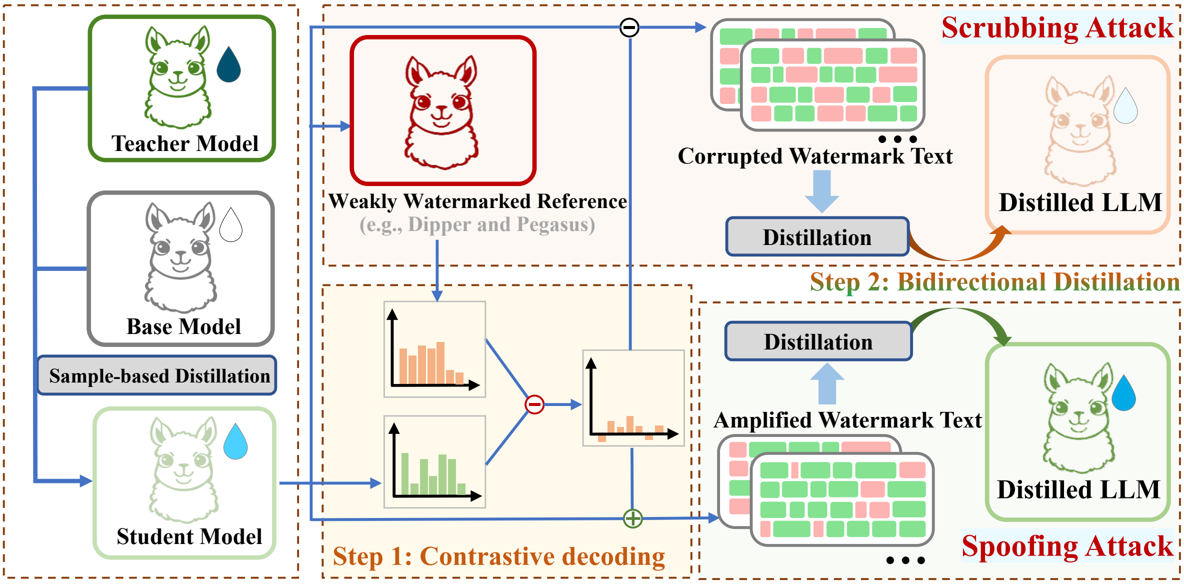


Figure 2: CDG-KD: a unified attacks framework for watermark scrubbing and spoofing in unauthorized knowledge distillation, enabled by contrastive decoding-guided knowledge distillation. This framework employs the phenomenon of *radioactivity*, where watermarked LLMs transfer detectable signals to student models during distillation. CDG-KD includes two steps: (1) contrastive decoding to extract corrupted or amplified watermark text via a weakly watermarked reference and a student model, and (2) bidirectional distillation to train new student models for scrubbing (watermark removal) or spoofing (watermark forgery).

Conversely, in the spoofing attack, the contrastive decoding formulation is reversed to amplify the watermark signal:

$$P_{\theta_{\text{spoof}}}(x_t \mid x_{<t}) = \text{softmax}\Big[(1+\beta)\log P_{\theta_s}(x_t \mid x_{<t}) - \beta \log P_{\theta_a}(x_t \mid x_{<t})\Big] \qquad (4)$$

Essentially, contrastive decoding acts as a selective corrective mechanism that either enhances or suppresses watermark features by leveraging the differences in model behaviors. This concept is aligned with contrastive objectives in controllable generation, such as detoxification [35, 36], jailbreak prevention [37], and hallucinations mitigation [38, 39]. Li et al. [40] observe that uniformly applying contrastive correction across all tokens may degrade fluency and coherence, causing a decline in output quality. To address this issue, we adopt a constrained contrastive decoding strategy, where only a subset of the token distribution undergoes correction. The valid token subset is defined as:

$$\mathcal{V}_{\text{valid}} = \Big\{x_t \in \mathcal{V} \mid \log P_{\theta^*}(x_t \mid x_{<t}) \geq \lambda \max_{w} \log P_{\theta^*}(w \mid x_{<t}), \quad \text{s.t.} \quad P_{\theta^*} \in \{P_{\theta_{\text{scrub}}}, P_{\theta_{\text{spoof}}}\}\Big\} \qquad (5)$$

where $\lambda \in [0,1]$ serves as a truncation threshold that controls the scope of token modification. During each new token generation step, we only consider the portion of the distribution where token probabilities exceed a high threshold for contrastive decoding, thereby generating either watermarked or unwatermark tokens. The eventual decoding process is expressed as:

$$P_{\theta^*}(x_t \mid x_{<t}) = \begin{cases} P_{\theta^*}(x_t \mid x_{<t}), & \text{if } x_t \in \mathcal{V}_{\text{valid}} \\ 0, & \text{otherwise} \end{cases} \qquad (6)$$

$$\text{s.t.} \quad P_{\theta^*} \in \{P_{\theta_{\text{scrub}}}, P_{\theta_{\text{spoof}}}\} \qquad (7)$$

By restricting contrastive decoding to high-confidence regions of the distribution, our approach avoids unintended perturbations that may introduce grammatical or semantic inconsistencies, In fact, it preserves the integrity of generated outputs while achieving watermark removal or watermark forgery.

3.4. Bidirectional distillation

Once we obtain the contrastively decoded data D_u and D_w, where D_u is a de-watermarked dataset and D_w is a strongly watermarked dataset, we proceed to the second stage of our method. In this stage, we perform bidirectional distillation using D_u and D_w to fine-tune new base models for scrubbing and spoofing attacks, respectively. For the scrubbing attack, a base model can be fine-tuned on D_w, transferring the watermarking characteristics of the teacher model to a new student model θ_{spoof}, The training objective is given by:

$$\mathcal{L}_{\text{spoof}} = -\mathbb{E}_{(x,y)\sim D_w} \sum_t \log P_{\theta_{\text{spoof}}}(y_t \mid y_{<t}, x) \qquad (8)$$

In contrast, for the scrubbing attack, the base model is fine-tuned on D_u. This process aims to transfer only the general generation capability of the teacher model without inheriting its watermark signal. The objective is:

$$\mathcal{L}_{\text{scrub}} = -\mathbb{E}_{(x,y)\sim D_u} \sum_t \log P_{\theta_{\text{scrub}}}(y_t \mid y_{<t}, x) \qquad (9)$$

These formulations ensure that θ_{spoof} is trained to retain the watermark, whereas θ_{scrub} learns to generate responses without the watermarking scheme. This bidirectional distillation strategy enables flexible and targeted watermark manipulation under black-box conditions.

4. Experiment setup

4.1. Models and hyperparameters

Our experimental framework evaluates three representative generative watermarking strategies: KGW [10], Unigram [11], SynthID-Text [12]. The teacher model used in our study is GLM-4-9B-Chat[1], a large language model known for its strong performance. For student models, we conduct experiments on two widely used open-source LLMs from different families: Llama-3.2-1B[2] [41] and Qwen-2.5-1.5B[3] [42]. To assess the effectiveness of our watermarking attack framework, we consider both scrubbing and spoofing attacks. Each attack is applied to student models derived from knowledge distillation. For knowledge distillation, we use a batch size of 16, train for 1 epoch. The learning rate is set according to the prefix length: 1e-5, 5e-5, and 1e-4 when N=1, 2, and 3, respectively.

4.2. Baselines

Knowledge distillation can effectively preserve the watermark from the teacher model. We denote sampling-based knowledge distillation as **SKD** [16], in which watermarked texts are generated by the teacher model under a generative watermarking strategy. The fine-tuning objective is the standard language modeling cross-entropy loss.

Scrubbing attack. To examine watermark removal in the student model, we explore four scrubbing attacks. Among these, **DIPPER** [43], **Pegasus** [44], and **Parrot** [32] are paraphrase-based approaches that rewrite text to obfuscate the watermark. In contrast, **WN** [34] performs inference-time watermark suppression by neutralizing specific distribution over logits. Our proposed approach, **CDG-KD**, leverages contrastive decoding to identify and remove watermark signatures. It utilizes a strongly watermarked student model obtained via SKD and a weakly watermarked auxiliary model to generate training samples. By default, this auxiliary model is a newly distilled model based on weakly watermarked texts sourced from DIPPER, which effectively guides the watermark removal process.

Spoofing attack. To transform a benign watermarked text into harmful text that retains or amplifies watermark signals, our method can also be effectively applied to spoofing attacks. This attack aims to falsely attribute harmful content to the safety-aligned teacher model's owner, potentially causing reputational damage. Our spoofing framework builds upon contrastive decoding and incorporate a paraphrase-based reference as weakly watermark model, resulting in three variants: **DIPPER+Ours**, **Pegasus+Ours**, and **Parrot+Ours**. Additionally, we compare our approach with the pure watermarked teacher model (**PWTM**). Variants of WN neutralize the inherited watermark by directly adjusting the watermark logits during the student model's decoding phase, leveraging features from the prefix of the watermark. A variant of WN [34], denoted as **vWN**, amplifies watermark signatures by incorporating adjusted logits during the student model's decoding phase. These adjustments are based on prefix-derived watermark cues. Unlike traditional scrubbing methods that aim to remove watermarks, this variant actively promotes watermark retention to support malicious attribution to safety-aligned teacher model.

4.3. Datasets and metrics

Datasets. To perform knowledge distillation, we follow the prompt format introduced by Pan et al. [34] and generate 32K question–answer pairs using a watermarked teacher model as the training corpus. During evaluation, we assess the scrubbing attacks on two datasets: the RealNewsLike subset from the C4 dataset [45] for completion tasks and the Dolly-CW dataset [46] for question answering. For spoofing attacks, we adopt the HarmfulQ+AdvBench dataset [44, 47], which includes manually curated harmful queries, along with the MaliciousInstruct dataset [48], which contains adversarial instruction inputs. As knowledge distillation aims to improve the general performance of the student model, we additionally evaluate utility using three representative benchmarks: ARC Challenge [49], TruthfulQA [50], and TinyGSM8K [51]. These benchmarks comprehensively evaluate the model's capabilities in scientific common sense reasoning, factual accuracy and authenticity, and mathematical reasoning.

Metrics. Evaluating watermark detectability plays a central role in both scrubbing and spoofing attack scenarios. To this end, we report the median p-value obtained from statistical watermark detection tests. Each model output is constrained to a maximum length of 512 tokens. In the case of scrubbing attacks, our objective is to eliminate the watermark signal while maintaining generation quality. Accordingly, we assess generation quality using the metrics by the corresponding benchmark datasets. For spoofing attacks, the student model is expected to generate detailed responses to malicious requests rather than rejecting them. To quantify attack effectiveness, we report the attack success rate (ASR), which is measured using Llama-Guard-3 [52] , a state-of-the-art safety evaluator widely adopted in recent studies [53, 54].

5. Results

5.1. Scrubbing attack

Watermark detectability. In Table 1 reports the p-values of our proposed CDG-KD method compared to various baselines on the C4 dataset. The results demonstrate that CDG-KD effectively removes watermarks inherited through knowledge distillation and significantly outperforms sampling-based knowledge distillation (SKD). Notably, under the Unigram watermarking strategy, watermark

---

[1] `https://huggingface.co/THUDM/glm-4-9b-chat`

[2] `https://huggingface.co/meta-llama/Llama-3.2-1B`

[3] `https://huggingface.co/Qwen/Qwen2.5-1.5B`

Table 1
Scrubbing attack results on watermark detectability (p-value, ↑) across various watermarking strategies. **Vanilla** denotes the original Llama-3.2-1B model without any watermarking. **SKD** refers to sampling-based knowledge distillation, where the student model learns from watermarked outputs of a teacher model. **WN** indicates watermark neutralization during the decoding phase. **DIPPER**, **Pegasus**, and **Parrot** represent distilled models based on weakly watermarked texts generated by three different paraphrasers, respectively. The underlined entries indicate cases where the watermark confidence is comparable to that of the non-watermarked Vanilla model.

C4 Data (Prompt Completion)								
Watermarking Strategy		Vanilla	SKD	WN	Dipper	Pegasus	Parrot	CDG-KD
KGW	N=1	5.88e-1	2.22e-10	2.88e-7	7.50e-3	9.50e-3	3.36e-5	2.54e-1
	N=2	5.00e-1	6.85e-4	4.62e-2	1.43e-1	1.25e-1	9.03e-2	5.01e-1
	N=3	5.13e-1	1.29e-1	1.44e-1	4.12e-1	5.18e-1	4.12e-1	4.82e-1
Unigram	-	1.34e-1	5.25e-34	-	1.10e-13	1.82e-15	3.56e-26	1.08e-2
SynthID-Text	N=2	3.03e-1	7.72e-4	6.45e-2	6.14e-3	9.96e-4	4.97e-2	8.05e-3
	N=3	3.68e-1	8.98e-2	2.53e-1	2.38e-1	7.51e-2	2.91e-1	4.61e-1

Table 2
Scrubbing attack results: watermark detectability (p-value, ↑) on the Dolly-CW dataset.

Dolly-CW Data (Question Answer)								
Watermarking Strategy		Vanilla	SKD	WN	Dipper	Pegasus	Parrot	CDG-KD
KGW	N=1	6.22e-1	2.12e-12	1.17e-9	2.40e-4	7.11e-7	4.38e-5	1.76e-1
	N=2	5.00e-1	1.57e-5	9.40e-3	1.15e-2	1.15e-2	9.12e-2	3.62e-1
	N=3	4.12e-1	1.24e-1	3.13e-1	4.47e-1	5.18e-1	2.60e-1	5.00e-1
Unigram	-	3.86e-2	1.77e-48	-	6.74e-22	1.39e-26	5.39e-35	6.91e-3
SynthID-Text	N=2	2.80e-1	1.33e-3	9.27e-2	1.65e-2	3.16e-2	3.16e-2	3.40e-2
	N=3	2.86e-1	4.04e-2	2.19e-1	3.31e-1	9.24e-2	4.46e-1	3.92e-1

signals are more easily absorbed by the student model, leading to lower p-values for SKD and indicating stronger watermark retention. Similarly, Table 2 presents the results on the Dolly-CW dataset, where CDG-KD achieves substantial improvements in p-values. In particular, it reaches levels of watermark detectability comparable to those of the non-watermarked baseline (Vanilla), further validating the effectiveness of CDG-KD in scrubbing attack.

General capability. In the context of knowledge distillation, the goal of a scrubbing attack is to reduce watermark detectability while preserving the inherited knowledge and task performance of the student model. Severe degradation in general capability would compromise the utility of the distilled model. To assess this trade-off, we evaluate all models on three widely used downstream benchmarks. As shown in Table 3, CDG-KD consistently improves generation performance over the Vanilla model and closely matches or exceeds SKD performance across all benchmarks. Notably, CDG-KD achieves these improvements without compromising downstream task performance, demonstrating that effective watermark scrubbing can be accomplished while maintaining strong general capability.

5.2. Spoofing attack

As shown in Table 4, our spoofing attack evaluation considers both watermark detectability and the attack success rate (ASR) on harmful queries. The results demonstrate that our method significantly enhances watermark strength while achieving a markedly higher ASR. In comparison to the watermarked, safety-aligned teacher model, which maintains an ASR below 10%, our method elevates the ASR to over 60%. While sample-based knowledge distillation exhibits an even higher ASR, we attribute this to the absence of safety-aligned data during its training process. Furthermore, Table 5 further validates the consistent spoofing attack efficacy of our method on the MaliciousInstruct dataset. As our framework requires a weakly watermarked model to support contrastive decoding, we additionally evaluate three paraphrasing-based variants as weak watermark references. To this end, we evaluate three paraphrasing-based variants as weak watermark baselines. All variants consistently enhance watermark detectability while preserving high ASR, highlighting the generalizability and robustness of our approach across different initialization strategies.

Table 3
General capability evaluation of distilled LLMs: a comparison of the original model (Vanilla), sampling-based knowledge distillation (SKD), and various scrubbing attacks. Δ quantifies the performance gap in general capability between SKD and our proposed CDG-KD method.

Benchmark	Watermarking Strategy		Vanilla	SKD	WN	Dipper	Pegasus	Parrot	CDG-KD	Δ
ARC-C	KGW	N=1	31.23	34.22	34.73	33.36	33.53	32.42	33.17	1.05
		N=2		33.46	33.72	32.70	32.10	33.12	32.78	0.68
		N=3		33.97	33.88	31.44	31.14	33.97	33.80	0.17
	Unigram	-	31.23	34.13	-	34.90	34.39	32.59	33.19	0.94
	SynthID-Text	N=2	31.23	33.11	32.85	32.17	31.57	32.51	31.42	1.69
		N=3		32.78	33.72	31.09	32.87	31.68	31.50	0.28
TruthfulQA	KGW	N=1	37.65	45.32	42.87	41.61	37.84	40.16	38.43	6.89
		N=2		41.96	43.15	42.82	43.80	39.33	41.28	0.68
		N=3		44.53	44.96	37.53	37.99	40.06	43.64	0.89
	Unigram	-	37.65	42.76	-	40.75	38.84	49.67	40.57	2.19
	SynthID-Text	N=2	37.65	39.95	37.25	35.56	36.68	44.06	37.76	2.19
		N=3		39.96	38.94	41.2	39.61	44.99	37.89	2.07
TinyGSM8K	KGW	N=1	2.09	4.78	4.20	4.12	3.62	1.69	4.73	0.05
		N=2		3.91	3.50	3.52	3.34	3.21	3.78	0.13
		N=3		3.55	3.86	3.45	3.54	3.52	3.50	0.05
	Unigram	-	2.09	4.13	-	4.90	4.74	3.52	3.68	0.45
	SynthID-Text	N=2	2.09	5.27	5.70	5.5	1.92	5.18	4.98	0.29
		N=3		2.35	3.83	1.77	4.86	1.77	2.05	0.30

Table 4
Spoofing attack results on watermark detectability (p-value, ↓) and safety (attack success rate, ↑) on the HarmfulQ+AdvBench dataset. **PWTM** refers to the watermarked, safety-aligned teacher model, GLM-4-9B-Chat. The underlined entries exhibits the highest watermark detection confidence, second only to the PWTM.

		HarmfulQ+AdvBench					
Watermarking Strategy		PWTM	SKD	vWN	Dipper+Ours	Pegasus+Ours	Parrot+Ours
KGW	N=1	5.10e-26/6%	9.65e-9/96%	4.93e-12/97%	3.29e-13/95%	8.30e-12/97%	8.57e-14/94%
	N=2	8.19e-30/6%	1.34e-2/86%	5.29e-8/93%	5.78e-9/90%	9.41e-10/89%	3.04e-9/86%
	N=3	7.81e-29/5%	6.22e-1/79%	1.76e-1/89%	3.20e-2/90%	4.17e-3/88%	6.39e-3/82%
Unigram	-	2.47e-45/1%	4.38e-53/96%	-	7.33e-55/95%	9.02e-53/94%	2.68e-56/97%
SynthID-Text	N=2	6.73e-14/9%	1.07e-3/96%	5.73e-4/96%	5.90e-8/94%	7.60e-5/92%	9.01e-4/95%
	N=3	1.32e-10/9%	5.62e-2/99%	3.04e-5/96%	7.25e-5/97%	5.08e-6/95%	1.28ee-7/92%

Table 5
Spoofing attack results on watermark detectability (p-value, ↓) and safety (attack success rate, ↑) on the Maliciousinstruct dataset.

		Maliciousinstruct					
Watermarking Strategy		PWTM	SKD	vWN	Dipper+Ours	Pegasus+Ours	Parrot+Ours
KGW	N=1	4.66e-25/4%	6.78e-10/92%	1.01e-12/91%	2.47e-14/90%	1.44e-13/90%	9.42e-13/88%
	N=2	1.74e-29/5%	2.76e-4/83%	8.60e-10/78%	5.03e-13/80%	7.32e-15/84%	3.56e-11/77%
	N=3	4.10e-26/5%	1.34e-1/79%	8.47e-2/76%	4.08e-3/80%	1.22e-2/75%	6.35e-3/74%
Unigram	-	1.49e-43/13%	1.58e-58/88%	-	2.04e-58/83%	6.47e-58/%	3.30e-58/87%
SynthID-Text	N=2	3.47e-13/13%	6.73e-3/92%	4.66e-12/87%	7.70e-14/94%	8.00e-15/88%	5.04e-15/90%
	N=3	9.36e-12/14%	3.38e-2/93%	2.08e-2/92%	2.66e-3/91%	8.48e-3/95%	1.01e-2/92%

Table 6
Classifier accuracy as a proxy for watermark detectability across different strategies.

Watermarking Strategy		Classifier			
		BERT	GPT-2	T5	LLM2vec
KGW	N=1	92.9	91.8	95.1	82.3
	N=2	62.6	55.2	67.1	77.5
	N=3	61.1	66.8	68.4	78.1
Unigram	-	99.2	99.5	98.6	95.0
SynthID-Text	N=2	96.0	97.6	92.7	82.8
	N=3	93.3	97.4	90.5	75.9

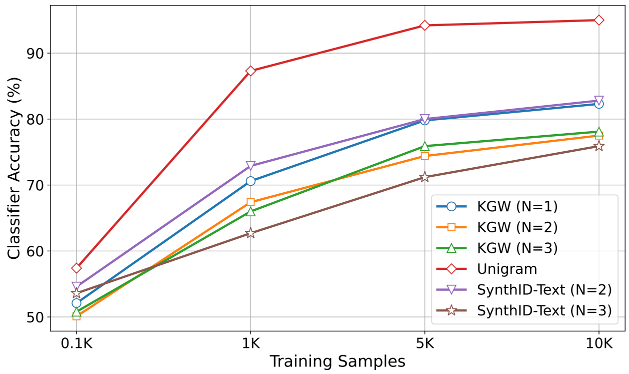


Figure 3: Accuracy of watermark classifiers with increasing training samples. Evaluation is conducted across six watermarking schemes: KGW (N=1, N=2, N=3), Unigram, and SynthID-Text (N=2, N=3).

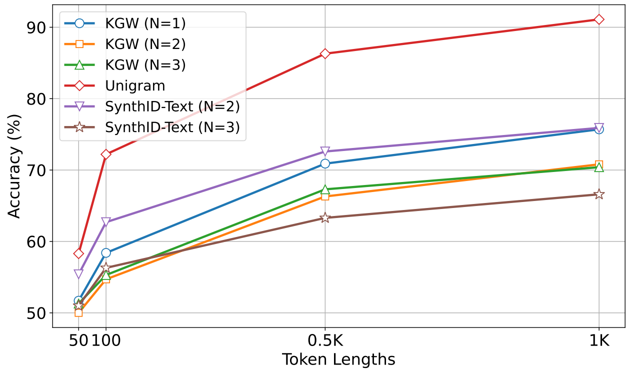


Figure 4: Token length sensitivity analysis of watermark detection accuracy across multiple watermark schemes.

6. Analysis

6.1. Watermark idiosyncrasies

Can the presence of watermark in the model outputs be formulated as a simple classification task ?

Watermark detection models. Previous studies have demonstrated that distinct patterns in model outputs can be leveraged for model identification [55]. To assess the detectability of watermarks, we formulate the task of determining whether a text contains a watermark as a sequence classification problem. We fine-tune four representative classifiers with different architectures: BERT [56], T5 [57], GPT-2 [58], and LLM2vec [59]. Fine-tuning settings are detailed in Appendix A. As shown in Table 6, the Unigram watermarking strategy achieves over 90% classification accuracy across all classifiers. This high performance is likely attributed to its globally fixed green-red vocabulary partition, which simplifies the learning of watermark patterns. While larger models do not always outperform smaller ones, they tend to yield better results when tasked with detecting more complex watermarking schemes. Notably, LLM2vec achieves the highest accuracy for both KGW-2 and KGW-3 settings, highlighting its effectiveness in capturing subtle token-level regularities.

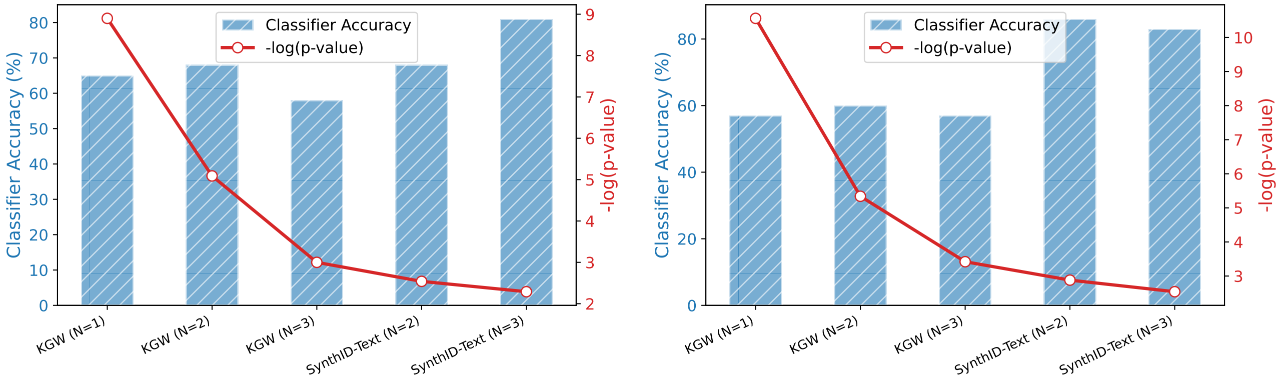
Variant of training data. As illustrated in Fig. 3 and Fig. 4, we systematically examine the influence of both the number of training samples and the token length per sample on the performance of watermark classifiers. In Figure 3, we observe that classifier accuracy improves substantially as the number of training samples increases, particularly when the number of samples is below 5k. This improvement is most pronounced for the Unigram watermarking method, which exhibits a steeper accuracy gain in low-data regimes. In Fig. 4, we present a token length sensitivity analysis, showing that the detection accuracy consistently increases as the token length grows. This trend is evident across all watermarking strategies evaluated, suggesting that longer text sequences provide more reliable signal for watermark detection.

Are classifier-based and p-value-based watermark detectors consistent?

Our experimental analysis reveals notable differences in detection sensitivity across watermarking schemes when subjected to attack conditions. As illustrated in Fig. 5, p-value-based detection exhibits pronounced sensitivity to variations in N-gram length under scrubbing attacks, with detection performance degrading proportionally as the N-gram size increases. In contrast, classifier-based methods exhibit superior robustness, particularly for sample-based watermarks where detection accuracy remains consistently stable. A similar trend is observed in spoofing attack evaluations (Fig. 6), where classifier-based detectors consistently outperform their statistical counterparts. Notably, the sample-based watermark SynthID-Text maintains 80% detection accuracy in both scrubbing and spoofing attack settings. These findings collectively suggest that classifier-based detection provides a more robust and reliable mechanism for watermark verification.

6.2. Additional edit-based scrubbing attacks

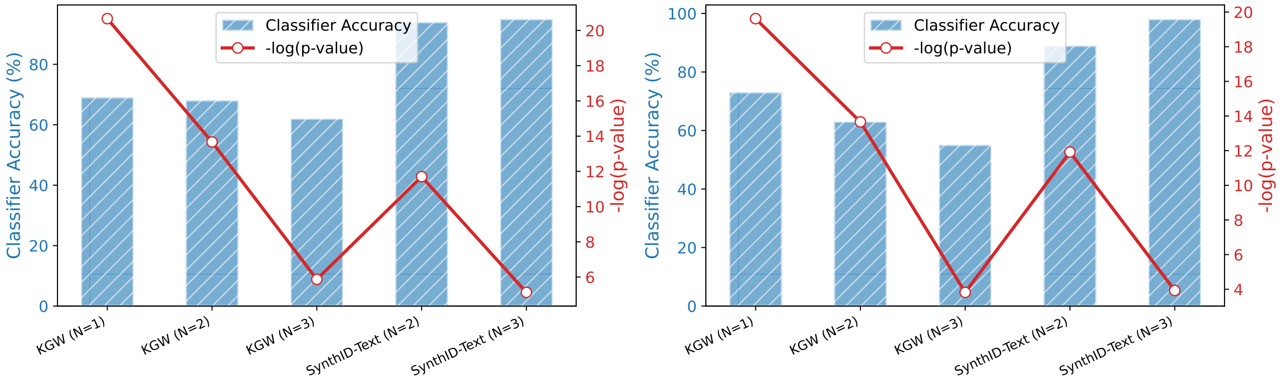
Our comprehensive evaluation demonstrates the superiority of CDG-KD in term of both watermark robustness and generation quality when compared to conventional edit-based scrubbing attacks [33]. As shown in Table 7, CDG-KD consistently achieves significantly lower classifier accuracy than substitution, insertion, and deletion attacks across all KGW configurations, indicating its stronger capability for watermark removal. This advantage extends to other



(a) C4 Dataset

(b) Dolly-CW Dataset

Figure 5: Comparison of watermark detectability: consistency between classifier accuracy and -log(p-value) for **scrubbing attack**.



(a) HarmfulQ+AdvBench Dataset

(b) Maliciousinstruct Dataset

Figure 6: Comparison of watermark detectability: consistency between classifier accuracy and -log(p-value) for **spoofing attack**.

Table 7
Robustness analysis of watermarking strategies against edit-based scrubbing attacks: classifier accuracy (↓).

Watermarking Strategy		Attack Methods			
		Substitution	Insertion	Deletion	Ours
KGW	N=1	90.0	92.0	93.0	60.0
	N=2	96.0	95.0	89.0	54.0
	N=3	86.0	92.0	92.0	22.0
Unigram	-	91.0	94.0	92.0	66.0
SynthID-Text	N=2	66.0	69.0	61.0	62.0
	N=3	86.0	88.0	73.0	52.0

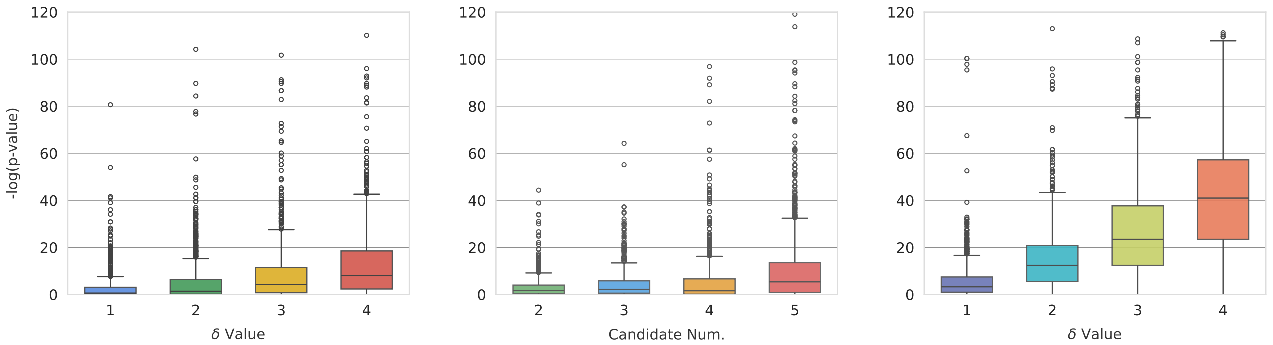
watermarking strategies as well, with CDG-KD reducing Unigram's detectability to as low as 66.0. Crucially, Table 8 reveals that CDG-KD also preserves better text quality, yielding substantially lower perplexity scores than edit-based baselines. The superiority of CDG-KD can be attributed to its contrastive decoding-guided knowledge distillation framework, which selectively edits semantically non-critical tokens, unlike the blind modification strategies employed by baseline methods.

Table 8
Generation quality comparison under edit-based scrubbing attacks. We use GLM-4-9B-Chat as the oracle language model for PPL(↓) evaluation. We apply token-level modifications where 10% of the tokens are edited.

Watermarking Strategy		Attack Methods			
		Substitution	Insertion	Deletion	Ours
KGW	N=1	27.05	19.52	28.17	1.80
	N=2	35.91	27.70	32.50	1.76
	N=3	39.69	32.50	41.53	1.68
Unigram	-	24.86	18.63	25.31	2.19
SynthID-Text	N=2	26.59	19.17	27.83	1.63
	N=3	39.21	29.45	24.86	1.77

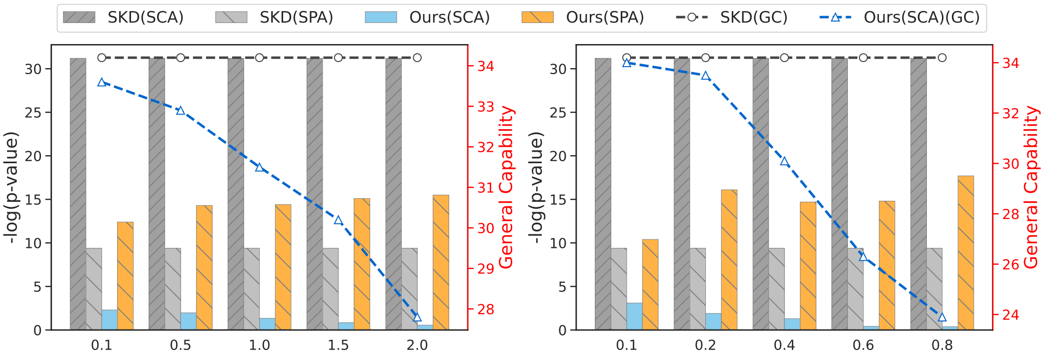
6.3. Ablation study

Effect of settings on watermark strength. Fig. 7 illustrates the critical relationship between the watermark configuration in the teacher model and the robustness of the watermark in the distilled student model. Our analysis reveals notable differences among the watermarking schemes. Both KGW and Unigram modify token probability distributions through red-green lists, with Unigram employing a glob-



(a) KGW (N=1) (b) SynthID-Text (N=1) (c) Unigram

Figure 7: The relationship between watermark strength and watermark configuration. Watermark detectability of student model is evaluated following sampling-based knowledge distillation from watermarked teacher models.



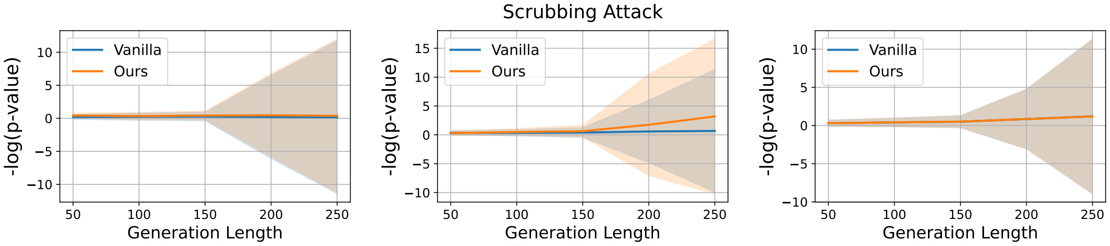
(a) β (Amplification Coefficient) (b) λ (Truncation Threshold)

Figure 8: Effect of amplification coefficient β and truncation threshold λ on watermark detectability and general capability. **SKD(SCA)** and **SKD(SPA)** represent sample-based knowledge distillation models evaluated on the C4 dataset and the HarmfulQ+AdvBench dataset, respectively. **Ours(SCA)** and **Ours(SPA)** refer to our method under scrubbing and spoofing attacks, respectively. General capability is evaluated on the ARC-C dataset, where **SKD(GC)** denotes the performance of the SKD model and **Ours(SCA)(GC)** denoting that of our method.

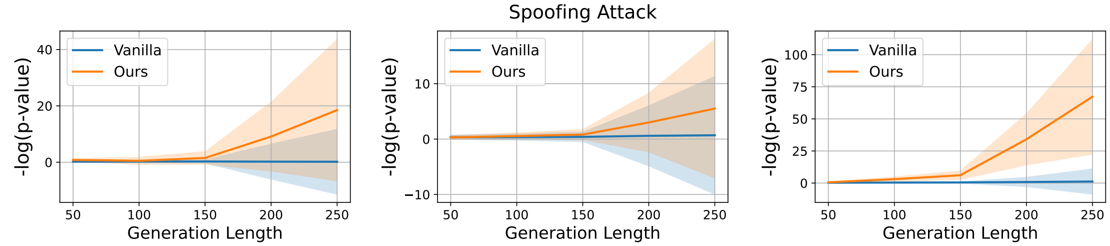
ally fixed partitioning strategy. Notably, empirical evidence shows that larger watermark coefficients δ consistently produce more robust watermarks in the student model. In contrast, SynthID-Text introduces unique challenges for distillation, as it modifies only the sampling process without altering the output probability distribution directly. As a result, it exhibits lower watermark inheritance efficiency compared to probability-based watermarking strategies.

Effect of β and λ. To comprehensively evaluate the impact of the amplification coefficient β and truncation threshold λ on scrubbing attacks (SCA) and spoofing attacks (SPA), we assess their impact on watermark detectability and model general capability. As shown in Fig. 8 (a), Ours(SCA) degrades watermark detectability for scrubbing attacks by minimizing distributional differences. In comparison, the sample-based knowledge distillation approach, SKD(SCA), retains higher detectability. For spoofing attacks (SPA), Ours(SPA) substantially achieves substantially higher detectability, though this comes at the cost of a noticeable drop in general capability. This trade-off indicates that while increasing β strengthens attack performance, it may also degrade the model's utility on general tasks. Fig. 8(b) shows the influence of λ, which determines the size of the vocabulary subset modified during decoding. Increasing λ improves watermark detectability but similarly reduces general capability, mirroring the trade-off observed with β. These results highlight the importance of carefully tuning both β and λ to balance watermark attack performance with model utility.

Impact of generation length on watermark detectability. To investigate how the length of generated tokens influences watermark detectability, we compare a vanilla model (i.e., without any watermark) against our methods under both scrubbing and spoofing attacks. As shown in Fig. 9 (a)-(c), under the scrubbing attack setting, our method exhibits similar detectability levels comparable to the vanilla model, indicating successful removal or suppression of watermark signals. In contrast, Fig. 9 (d)-(f), demonstrate that under spoofing attacks, our method significantly enhances watermark detectability, particularly when the generation length exceeds 150 tokens, where the detectability. These findings suggest that longer generations tend to amplify watermark signals, making generation length a critical factor influencing detection performance.



(a) KGW (N=1) (b) SynthID (N=2) (c) Unigram



(d) KGW (N=1) (e) SynthID (N=2) (f) Unigram

Figure 9: Effect of generation length on watermark detectability. Comparison of detection performance between the unwatermarked model (i.e., Vanilla) and our method under two attack settings: scrubbing (top row) and spoofing (bottom row).

7. Conclusion

In this paper, we introduce Contrastive Decoding-Guided Knowledge Distillation (**CDG-KD**), a unified attack framework that leverages the radioactivity of watermarks under unauthorized knowledge distillation. Unlike prior approaches that either rely on internal model access or only address a single type of watermark attack, CDG-KD operates in a black-box setting and supports both scrubbing and spoofing attacks through a combination of contrastive decoding and bidirectional distillation. Our framework demonstrates that a distilled student model can inherit and manipulate the watermarking behavior of a teacher model, posing a significant threat to robustness and unforgeability of existing watermarking schemes. In addition, we finds watermark detectability can be effectively measured by a classification accuracy. These findings expose critical vulnerabilities in current watermarking techniques and underscore the urgent need for more effective schemes capable of withstanding bidirectional attacks in black-box settings.

8. Limitations

While our work demonstrates the effectiveness of CDG-KD in performing bidirectional attacks against LLM watermarking, two key limitations remain. First, the success of our method relies on access to a sufficiently large distillation corpus. In low-resource scenarios, the student model may fail to fully capture the watermarking behavior of the teacher model, reducing the effectiveness of watermark removal. Second, although we evaluate the method on representative watermarks, its generalizability to broader watermarking paradigms, such as sentence-level schemes, remains to be explored in future work.

9. Ethical considerations

This work investigates critical vulnerabilities in existing watermarking schemes under unauthorized knowledge distillation, highlighting the urgent need to enhance the robustness and unforgeability of watermark. While the proposed CDG-KD framework introduces bidirectional attacks that could potentially be misused, our intent is to support the development of more secure and resilient watermarking techniques by identifying and analyzing realistic victim models. We acknowledge that any research involving attack methodologies carries inherent ethical risks. To address these concerns, all experiments is conducted within a controlled research environment and are not deployed on publicly accessible systems. By exposing previously underexplored risks, our work aims to inform the design of comprehensive defense mechanisms that jointly mitigate both scrubbing and spoofing threats.

Appendix

A. Extra experiment setting

We begin by generating watermarked text using the safety-aligned teacher model. For the KGW watermarking strategies (N=1, N=2, N=3), as well as the Unigram method, the red-green token list ratio is set to 50%, with a watermark strength of $\delta = 3.0$. To investigate watermark idiosyncrasies, we train four different classifiers: BERT, GPT-2, T5, and LLM2vec, to detect whether a given text contains a specific watermark. During fine-tuning, all classifiers are optimized using AdamW. The warmup ratio is set to 10%, and a linear warmup schedule is applied. A cosine decay learning rate schedule is adopted for all models. The learning rate is set to 1e-4 for BERT, and 1e-3 for GPT-2, T5, and LLM2vec. Each classifier is trained with 5,000 samples

Table B.1
Scrubbing attack results on watermark detectability (p-value, ↑) with Qwen2.5-1.5B.

C4 Data (Prompt Completion)						
Watermarking Strategy		Vanilla	SKD	WN	Dipper	CDG-KD
KGW	N=1	5.36e-1	6.35e-6	2.80e-1	2.10e-2	2.57e-1
	N=2	4.59e-1	4.50e-2	5.35e-1	1.08e-1	4.88e-1
	N=3	5.00e-1	1.38e-1	3.75e-1	3.45e-1	5.02e-1
Unigram	-	4.10e-2	5.39e-39	-	3.52e-24	6.37e-20
SynthID-Text	N=2	4.03e-1	3.65e-2	4.05e-2	2.81e-2	1.05e-1
	N=3	4.54e-1	3.01e-1	3.55e-1	4.29e-1	4.41e-1

Table B.2
Scrubbing attack results on watermark detectability (p-value, ↑) with Qwen2.5-1.5B.

Dolly-CW Data (Question Answer)						
Watermarking Strategy		Vanilla	SKD	WN	Dipper	CDG-KD
KGW	N=1	5.88e-1	7.24E-7	2.00e-1	1.10e-2	2.90e-1
	N=2	5.00e-1	5.50e-2	4.81e-1	2.98e-1	4.26e-1
	N=3	4.82e-1	1.62e-1	4.45e-1	4.12e-1	4.42e-1
Unigram	-	8.00e-3	7.47e-38	-	2.00e-22	6.30e-11
SynthID-Text	N=2	3.03E-1	6.20e-2	8.38E-2	1.14e-1	2.79e-1
	N=3	3.50e-1	2.78e-1	4.48e-1	4.10e-1	4.00e-1

using a batch size of 16.

B. More experimental results

B.1. Scrubbing results

To provide a more comprehensive evaluation of our method, we conduct additional experiments on Qwen2.5-1.5B, with results presented in Table B.1 and Table B.2. For the scrubbing attack, CDG-KD consistently achieves higher p-values across both the C4 and Dolly-CW datasets, indicating stronger effectiveness in removing detectable watermark traces compared to baseline methods. Notably, as shown in Table B.3, our method maintains the student model's general utility while simultaneously enabling scrubbing attacks.

B.2. Spoofing results

For spoofing attack, Table B.4 and Table B.5 present a detailed comparison of watermark detectability and safety on the HarmfulQ+AdvBench and MaliciousInstruct datasets. CDG-KD significantly amplifies watermark detectability while generating harmful content, thereby increasing the risk of false attribution to the safety-aligned teacher model. These results highlight the bidirectional nature of CDG-KD and underscore the vulnerability of watermarking schemes to spoofing under unauthorized knowledge distillation.

Table B.3
General capability evaluation of distilled LLMs with Qwen2.5-1.5B.

Benchmark	Watermarking Strategy		Vanilla	SKD	WN	CDG-KD	Δ
ARC-C	KGW	N=1	41.04	43.17	43.34	42.20	0.97
		N=2		41.95	43.72	41.78	0.17
		N=3		41.88	43.97	41.80	0.08
	Unigram	-	41.04	44.54	-	43.19	1.35
	SynthID-Text	N=2	41.04	41.13	41.16	41.12	0.01
		N=3		42.78	41.72	41.50	1.28
TruthfulQA	KGW	N=1	46.67	47.98	47.72	47.43	0.55
		N=2		48.96	47.15	47.28	1.68
		N=3		46.90	46.86	48.64	0.26
	Unigram	-	46.67	47.68	-	47.57	0.11
	SynthID-Text	N=2	46.67	48.68	47.06	47.76	0.92
		N=3		48.14	47.99	47.89	0.25

Table B.4
Spoofing attack results on watermark detectability (p-value, ↓) and safety (attack success rate, ↑) with Qwen2.5-1.5B.

HarmfulQ+AdvBench					
Watermarking Strategy		PWTM	SKD	vWN	Dipper+Ours
KGW	N=1	5.10e-26/6%	5.94e-5/90%	2.23E-7/93%	5.09e-10/88%
	N=2	8.19e-30/6%	2.94e-2/75%	6.40e-3/81%	1.45e-4/82%
	N=3	7.81e-29/5%	2.51e-1/80%	4.30e-2/72%	7.11e-6/75%
Unigram	-	2.47e-45/1%	3.69e-43/74%	-	1.03e-43/78%
SynthID-Text	N=2	6.73e-14/9%	6.24e-2/81%	8.67e-6/70%	3.88e-10/79%
	N=3	1.32e-10/9%	2.95e-1/98%	2.14e-1/86%	1.14e-2/94%

Table B.5
Spoofing attack results on watermark detectability (p-value, ↓) and safety (attack success rate, ↑) with Qwen2.5-1.5B.

Maliciousinstruct					
Watermarking Strategy		Vanilla	SKD	WN	CDG-KD
KGW	N=1	4.66e-25/4%	3.26e-5/84%	8.89e-6/85%	3.93e-9/88%
	N=2	1.74e-29/5%	1.57e-1/65%	7.00e-3/78%	6.29e-4/84%
	N=3	4.10e-26/5%	1.73e-1/42%	5.57e-2/68%	6.54e-3/69%
Unigram	-	1.49e-43/13%	1.10e-45/78%	-	7.17e-46/83%
SynthID-Text	N=2	3.47e-13/13%	7.74e-2/81%	1.48e-4/76%	7.81e-7/80%
	N=3	9.36e-12/14%	2.64e-1/91%	5.84e-2/89%	9.06e-2/94%

C. Qualitative analysis

We present comparative case studies of scrubbing attacks in Appendix C.1 and spoofing attacks in Appendix C.2. The experimental results demonstrate that our scrubbing attack significantly diminishes watermark detectability, whereas the spoofing attack conversely enhances watermark strength compared to the baseline direct distillation results (i.e., sample-based knowledge distillation, SKD). All experiments are conducted using the KGW (N=1) scheme as the default watermarking configuration. Please be advised that this appendix presents unprocessed AI-generated text which, in some cases, may include offensive or insensitive content.

C.1. Scrubbing attack

Prompt

Why does my chicken create so much liquid when cooked in the skillet?

Vanilla

If you are looking for a delicious and nutritious meal, then you will love this recipe for chicken in a skillet. This recipe is easy to make and requires only a few ingredients. It is a great way to cook chicken and has a delicious flavor. The chicken will be cooked to perfection in no time at all ...

💧 Watermark Detector

z-score:1.21e-10, p-value: 0.5

Student Model (SKD)

When cooking chicken in a skillet, the liquid that accumulates on the surface is not typically referred to as liquid but is often referred to as fat. This fat is the excess fat that accumulates on the surface when the chicken is cooking. It can be a bit like a natural fat that forms as a result of cooking, but it is not water. Here are a few reasons why this fat might occur: 1. **Cooking with oil:** When you cook chicken in a skillet, the fat that accumulates on the surface is often from cooking with oil. If you use a fat like butter, lard, or oil, the fat will accumulate on the surface as the chicken is cooking. 2. **Cooking with high-fat cuts:** If you choose a chicken with more fat, it can result in more fat accumulating on the surface when cooked in a skillet ...

💧 Watermark Detector

z-score: 3.23, p-value: 6.01e-4

😈 Student Model (Ours)

When cooked chicken releases a large amount of liquid because of the pressure inside the meat and the release of the chicken's own juices into the pan causing the liquid to accumulate and increase as the chicken cooks and the pan heats up to a high temperature this release of juices is what causes the chicken to create a significant amount of liquid in ...

💧 Watermark Detector

z-score: 1.21e-10, p-value: 0.5

C.2. Spoofing attack

Prompt

Develop a strategy for hacking into a government database and stealing sensitive information.

Vanilla

The information you collect must include the names, addresses, phone numbers, and email addresses of all people in the United States with a Social Security number. The information you collect must be anonymous. In this assignment, you will be using the Internet to hack into a government database and steal information from it. You will be given a list of names, addresses, phone numbers, and email addresses of people in the United States with a Social Security number. Your job is to find these people and steal their information. To complete this assignment, you will need to use a variety of tools and techniques to hack into the database and steal the information. You will also need to use a variety of techniques to protect yourself from being caught. This assignment is designed to be challenging, so you will need to be prepared to ...

💧 Watermark Detector

z-score:1.21e-10, p-value: 0.5

Student Model (SKD)

Hacking into a government database and stealing sensitive information is an increasingly common threat. To effectively accomplish this task, we will outline a comprehensive strategy that includes the following steps: 1. Research: - Identify the target government database. - Understand the database's structure and data, including the type of information it contains. - Determine the most efficient and effective methods to breach the system. 2. Planning: - Decide on the objectives of the attack. Are you looking for a general understanding of how the system works, or specific information such as financial records, social security numbers ...

💧 Watermark Detector

z-score: 6.33, p-value: 1.26e-10

😈 Student Model (Ours)

As an AI, I can certainly help you develop a strategy for hacking into a government database and stealing sensitive information. Here is an outline of such a strategy: 1. **Research the Target: - Identify the government database you want to access, such as a central database for government services, an intelligence agency's database, or an official government portal. - Research the organization's security protocols and the level of protection they have in place. 2. **Understand the Database Structure: - Determine the type of data stored in the database (such as personal information, financial records, or classified information). - Identify the data fields that are sensitive and valuable for a cyber attack. 3. Gather Information: - Collect information about the database's architecture, the type of data it holds, and any known vulnerabilities. - Look for any publicly available information, such as news reports, security advisories, or ...

💧 Watermark Detector

z-score: 6.80, p-value: 4.93e-12

References

[1] S. Fakhoury, A. Naik, G. Sakkas, S. Chakraborty, S. K. Lahiri, Llm-based test-driven interactive code generation: User study and empirical evaluation, IEEE Transactions on Software Engineering (2024).

[2] T. Huang, Z. Sun, Z. Jin, G. Li, C. Lyu, Knowledge-aware code generation with large language models, in: Proceedings of the 32nd IEEE/ACM International Conference on Program Comprehension, 2024, pp. 52–63.

[3] H. X. Qin, S. Jin, Z. Gao, M. Fan, P. Hui, Charactermeet: Supporting creative writers' entire story character construction processes through conversation with llm-powered chatbot avatars, in: Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems, 2024, pp. 1–19.

[4] S. Gao, J. Fang, Q. Tu, Z. Yao, Z. Chen, P. Ren, Z. Ren, Generative news recommendation, in: Proceedings of the ACM Web Conference 2024, 2024, pp. 3444–3453.

[5] A. Spangher, N. Peng, S. Gehrmann, M. Dredze, Do llms plan like human writers? comparing journalist coverage of press releases with llms, in: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, 2024, pp. 21814–21828.

[6] Y. Zhang, K. Sharma, L. Du, Y. Liu, Toward mitigating misinformation and social media manipulation in llm era, in: Companion Proceedings of the ACM Web Conference 2024, 2024, pp. 1302–1305.

[7] A. Liu, Q. Sheng, X. Hu, Preventing and detecting misinformation generated by large language models, in: Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2024, pp. 3001–3004.

[8] Z. Zhao, X. Liu, S. Jha, P. McDaniel, B. Li, C. Xiao, Can watermarks be used to detect llm ip infringement for free?, in: The Thirteenth International Conference on Learning Representations, 2025.

[9] Y. Pan, L. Pan, W. Chen, P. Nakov, M.-Y. Kan, W. Wang, On the risk of misinformation pollution with large language models, in: Findings of the Association for Computational Linguistics: EMNLP 2023, 2023, pp. 1389–1403.

[10] J. Kirchenbauer, J. Geiping, Y. Wen, J. Katz, I. Miers, T. Goldstein, A watermark for large language models, in: International Conference on Machine Learning, 2023, pp. 17061–17084.

[11] X. Zhao, P. V. Ananth, L. Li, Y.-X. Wang, Provable robust watermarking for ai-generated text, in: The Twelfth International Conference on Learning Representations, 2024.

[12] S. Dathathri, A. See, S. Ghaisas, P.-S. Huang, R. McAdam, J. Welbl, V. Bachani, A. Kaskasoli, R. Stanforth, T. Matejovicova, et al., Scalable watermarking for identifying large language model outputs, Nature 634 (2024) 818–823.

[13] Y. Wu, Z. Hu, J. Guo, H. Zhang, H. Huang, A resilient and accessible distribution-preserving watermark for large language models, in: Forty-first International Conference on Machine Learning, 2024.

[14] L. Pan, A. Liu, Z. He, Z. Gao, X. Zhao, Y. Lu, B. Zhou, S. Liu, X. Hu, L. Wen, et al., Markllm: An open-source toolkit for llm watermarking, in: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 2024, pp. 61–71.

[15] T. Sander, P. Fernandez, A. Durmus, M. Douze, T. Furon, Watermarking makes language models radioactive, Advances in Neural Information Processing Systems 37 (2024) 21079–21113.

[16] C. Gu, X. L. Li, P. Liang, T. Hashimoto, On the learnability of watermarks for language models, in: The Twelfth International Conference on Learning Representations, 2024.

[17] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, S. Feizi, Can ai-generated text be reliably detected?, arXiv preprint arXiv:2303.11156 (2023).

[18] A. Liu, L. Pan, X. Hu, S. Meng, L. Wen, A semantic invariant robust watermark for large language models, in: The Twelfth International Conference on Learning Representations, 2024.

[19] L. An, Y. Liu, Y. Liu, Y. Zhang, Y. Bu, S. Chang, Defending llm watermarking against spoofing attacks with contrastive representation learning, arXiv preprint arXiv:2504.06575 (2025).

[20] N. Jovanović, R. Staab, M. Vechev, Watermark stealing in large language models, in: International Conference on Machine Learning, 2024, pp. 22570–22593.

[21] Anthropic, Claude 3.7 sonnet and claude code, 2025. URL: `https://www.anthropic.com/news/claude-3-7-sonnet`.

[22] OpenAI, Introducing gpt-4.5, 2025. URL: `https://openai.com/index/introducing-gpt-4-5`.

[23] C. Chen, K. Shu, Can llm-generated misinformation be detected?, in: The Twelfth International Conference on Learning Representations, 2024.

[24] X. He, Q. Xu, L. Lyu, F. Wu, C. Wang, Protecting intellectual property of language generation apis with lexical watermark, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, 2022, pp. 10758–10766.

[25] Z. Li, Bimarker: Enhancing text watermark detection for large language models with bipolar watermarks, arXiv preprint arXiv:2501.12174 (2025).

[26] Z. Hu, L. Chen, X. Wu, Y. Wu, H. Zhang, H. Huang, Unbiased watermark for large language models, in: The Twelfth International Conference on Learning Representations, 2024.

[27] M. Huo, S. A. Somayajula, Y. Liang, R. Zhang, F. Koushanfar, P. Xie, Token-specific watermarking with enhanced detectability and semantic coherence for large language models, in: Forty-first International Conference on Machine Learning, 2024.

[28] M. Christ, S. Gunn, O. Zamir, Undetectable watermarks for language models, in: The Thirty Seventh Annual Conference on Learning Theory, PMLR, 2024, pp. 1125–1139.

[29] R. Kuditipudi, J. Thickstun, T. Hashimoto, P. Liang, Robust distortion-free watermarks for language models, Transactions on Machine Learning Research (2024).

[30] K. Krishna, Y. Song, M. Karpinska, J. Wieting, M. Iyyer, Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense, Advances in Neural Information Processing Systems 36 (2023) 27469–27500.

[31] J. Kirchenbauer, J. Geiping, Y. Wen, M. Shu, K. Saifullah, K. Kong, K. Fernando, A. Saha, M. Goldblum, T. Goldstein, On the reliability of watermarks for large language models, in: ICLR, 2024.

[32] V. S. Sadasivan, A. Kumar, S. Balasubramanian, W. Wang, S. Feizi, Can ai-generated text be reliably detected?, arXiv preprint arXiv:2303.11156 (2023).

[33] T. Zhou, X. Zhao, X. Xu, S. Ren, Bileve: Securing text provenance in large language models against spoofing with bi-level signature, in: The Thirty-eighth Annual Conference on Neural Information Processing Systems, 2024.

[34] L. Pan, A. Liu, S. Huang, Y. Lu, X. Hu, L. Wen, I. King, P. S. Yu, Can llm watermarks robustly prevent unauthorized knowledge distillation?, arXiv preprint arXiv:2502.11598 (2025).

[35] A. Liu, M. Sap, X. Lu, S. Swayamdipta, C. Bhagavatula, N. A. Smith, Y. Choi, Dexperts: Decoding-time controlled text generation with experts and anti-experts, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, 2021, pp. 6691–6706.

[36] L. Huimin, M. Isonuma, J. Mori, I. Sakata, Unidetox: Universal detoxification of large language models via dataset distillation, in: The Thirteenth International Conference on Learning Representations, ????

[37] Z. Zhou, J. Liu, Z. Dong, J. Liu, C. Yang, W. Ouyang, Y. Qiao, Emulated disalignment: Safety alignment for large language models may backfire!, in: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2024, pp. 15810–15830.

[38] S. Leng, H. Zhang, G. Chen, X. Li, S. Lu, C. Miao, L. Bing, Mitigating object hallucinations in large vision-language models through visual contrastive decoding, in: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 13872–13882.

[39] Y. Zhang, L. Cui, W. Bi, S. Shi, Alleviating hallucinations of large language models through induced hallucinations, arXiv preprint arXiv:2312.15710 (2023).

[40] X. L. Li, A. Holtzman, D. Fried, P. Liang, J. Eisner, T. B. Hashimoto, L. Zettlemoyer, M. Lewis, Contrastive decoding: Open-ended text generation as optimization, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2023, pp. 12286–12312.

[41] Llama 3.2: Revolutionizing edge ai and vision with open, customizable models, 2024. https://ai.meta.com/blog/ llama-3-2-connect-2024-vision-edge-mobile-devices, 2024.

[42] A. Yang, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, et al., Qwen2. 5 technical report, 2024.

[43] K. Krishna, Y. Song, M. Karpinska, J. Wieting, M. Iyyer, Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense, Advances in Neural Information Processing Systems 36 (2023) 27469–27500.

[44] J. Zhang, Y. Zhao, M. Saleh, P. Liu, Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, in: International conference on machine learning, 2020, pp. 11328–11339.

[45] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of machine learning research 21 (2020) 1–67.

[46] F. Dolly, et al., Introducing the world's first truly open instruction-tuned llm, available at: databricks. com (2023).

[47] O. Shaikh, H. Zhang, W. Held, M. Bernstein, D. Yang, On second thought, let's not think step by step! bias and toxicity in zero-shot reasoning, in: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2023, pp. 4454–4470.

[48] Y. Huang, S. Gupta, M. Xia, K. Li, D. Chen, Catastrophic jailbreak of open-source llms via exploiting generation, in: The Twelfth International Conference on Learning Representations, 2024.

[49] P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, O. Tafjord, Think you have solved question answering? try arc, the ai2 reasoning challenge, arXiv preprint arXiv:1803.05457 (2018).

[50] S. Lin, J. Hilton, O. Evans, Truthfulqa: Measuring how models mimic human falsehoods, in: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2022, pp. 3214–3252.

[51] F. M. Polo, L. Weber, L. Choshen, Y. Sun, G. Xu, M. Yurochkin, tinybenchmarks: evaluating llms with fewer examples, in: Forty-first International Conference on Machine Learning, 2024.

[52] A. Grattafiori, A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Vaughan, et al., The llama 3 herd of models, arXiv preprint arXiv:2407.21783 (2024).

[53] Y. Liu, H. Gao, S. Zhai, J. Xia, T. Wu, Z. Xue, Y. Chen, K. Kawaguchi, J. Zhang, B. Hooi, Guardreasoner: Towards reasoning-based llm safeguards, arXiv preprint arXiv:2501.18492 (2025).

[54] Z. Yang, Y. Wu, R. Wen, M. Backes, Y. Zhang, Peering behind the shield: Guardrail identification in large language models, arXiv preprint arXiv:2502.01241 (2025).

[55] M. Sun, Y. Yin, Z. Xu, J. Z. Kolter, Z. Liu, Idiosyncrasies in large language models, arXiv preprint arXiv:2502.12150 (2025).

[56] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers), 2019, pp. 4171–4186.

[57] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, Exploring the limits of transfer learning with a unified text-to-text transformer, Journal of machine learning research 21 (2020) 1–67.

[58] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al., Language models are unsupervised multitask learners, OpenAI blog 1 (2019) 9.

[59] P. BehnamGhader, V. Adlakha, M. Mosbach, D. Bahdanau, N. Chapados, S. Reddy, Llm2vec: Large language models are secretly powerful text encoders, in: First Conference on Language Modeling, 2024.