CATER: Intellectual Property Protection on Text Generation APIs via Conditional Watermarks

Xuanli He[*]
University College London
zodiac.he@gmail.com

Qiongkai Xu [*]
University of Melbourne
qiongkai.xu@unimelb.edu.au

Yi Zeng[†]
Virginia Tech
yizeng@vt.edu

Lingjuan Lyu[‡]
Sony AI
Lingjuan.Lv@sony.com

Fangzhao Wu
Microsoft Research Asia
fangzwu@microsoft.com

Jiwei Li
Shannon.AI, Zhejiang University
jiwei_li@shannonai.com

Ruoxi Jia
Virginia Tech
ruoxijia@vt.edu

Abstract

Previous works have validated that text generation APIs can be stolen through imitation attacks, causing IP violations. In order to protect the IP of text generation APIs, recent work has introduced a watermarking algorithm and utilized the null-hypothesis test as a post-hoc ownership verification on the imitation models. However, we find that it is possible to detect those watermarks via sufficient statistics of the frequencies of candidate watermarking words. To address this drawback, in this paper, we propose a novel Conditional wATERmarking framework (CATER) for protecting the IP of text generation APIs. An optimization method is proposed to decide the watermarking rules that can minimize the distortion of overall word distributions while maximizing the change of conditional word selections. Theoretically, we prove that it is infeasible for even the savviest attacker (they know how CATER works) to reveal the used watermarks from a large pool of potential word pairs based on statistical inspection. Empirically, we observe that high-order conditions lead to an exponential growth of suspicious (unused) watermarks, making our crafted watermarks more stealthy. In addition, CATER can effectively identify IP infringement under architectural mismatch and cross-domain imitation attacks, with negligible impairments on the generation quality of victim APIs. We envision our work as a milestone for stealthily protecting the IP of text generation APIs.

1 Introduction

Nowadays, many technology corporations, such as Google, Amazon, Microsoft, have invested a plethora of workforce and computation to data collection and model training, in order to deploy well-trained commercial models as pay-as-you-use services on their cloud platforms. Therefore, these corporations own the intellectual property (IP) of their trained models. Unfortunately, previous works have validated that the functionality of a victim API can be stolen through imitation attacks, which

[*] Equal contribution. Most of the work was finished when X.H was at Monash Unversity.

[†] Work done during internship at Sony AI.

[‡] Corresponding author.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

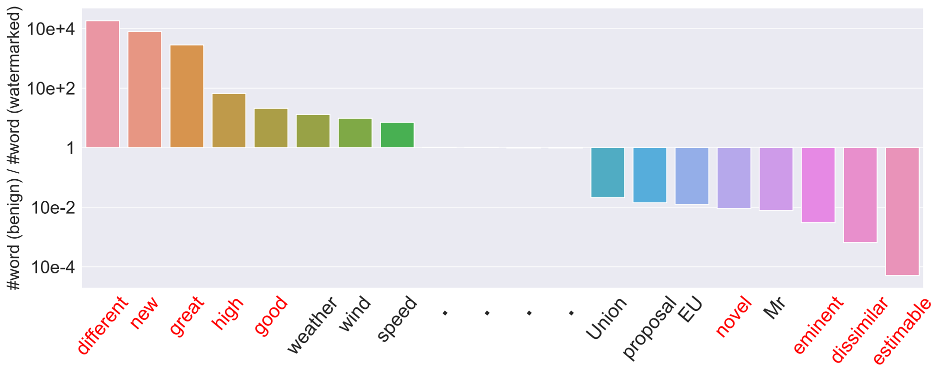


Figure 1: Ratio change of word frequency of top 100 words between benign and watermarked corpora used by [13], namely $P_b(w)/P_w(w)$. Red words are the selected watermarks. Although we only list 16 words having the most significant ratio change in the benign and watermarked corpora and omit the rest of them for better visualization, all watermarks are within the top 100 words.

inquire the victim with carefully designed queries and train an imitation model based on the outputs of the target API. Such attacks cause severe IP violations of the target API and stifle the creativity and motivation of our research community [44, 48, 20, 12, 9].

In fact, imitation attacks work not only on laboratory models, but also on commercial APIs [48, 51], since the enormous commercial benefit allures competing companies or individual users to extract or steal these successful APIs. For instance, some leading companies in NLP business have been caught imitating their competitors' models [40]. Beyond imitation attacks, the attacker could potentially surpass victims by conducting unsupervised domain adaptation and multi-victim ensemble [51].

In order to protect victim models, He et al. [13] first introduced a watermarking algorithm to text generation and utilized the null-hypothesis test as a post-hoc ownership verification on the imitation models. However, traditional watermarking methods generally distort the word distribution, which could be utilized by attackers to infer the watermarked words via sufficient statistics of the frequency change of candidate watermarking words. As an example shown in Figure 1, the replaced words and their substitutions are those with most frequency decrease ratios and increase ratios, respectively. To address this drawback, we are motivated to develop a more stealthy watermarking method to protect the IP of text generation APIs. The stealthiness of the new watermarks is achieved by incorporating high-order linguistic features as conditions that trigger corresponding watermarking rules.

Overall, our main contributions are as follows[4]:

- We propose a novel Conditional wATERmarking framework (CATER) for protecting text generation APIs. An optimization method is proposed to decide the watermarking rules that can *i)* minimize the distortion of overall word distributions, while *ii)* maximize the change of conditional word selections.
- Theoretically, we prove that a small number of the used watermarks could be blended in and camouflaged by a large number of suspicious watermarks when attackers attempt to inverse the backend watermarking rules.
- Empirically, we observe that high-order conditions lead to the exponential growth of suspicious (unused) watermarks, which encourages better stealthiness of the proposed defense with little hurt to the generation of victim models.

2 Preliminary and Background

2.1 Imitation Attack

An imitation attack (a.k.a model extraction) aims to emulate the behavior of the victim model $\mathcal{V}$, such that the adversary can either sidestep the service charges or launch a competitive service [44, 20, 48, 12, 51]. Malicious users can achieve this goal through interaction with the victim model $\mathcal{V}$ without knowing its internals, such as the model architecture, hyperparameters, training data, *etc.* Adversaries

[4] Code and data are available at: `https://github.com/xlhex/cater_neurips.git`

first craft a set of queries Q based on the documentation of a target model. Then Q will be sent to $\mathcal{V}$ to obtain the corresponding predictions Y. Finally, an imitation model $\mathcal{S}$ can be attained by learning a function to map Q to Y.

Most prior imitation attacks are limited to classification tasks [44, 30, 12]. The imitation for text generation, a crucial task in natural language processing, has been under-developed until recently. Inspired by the efficacy of sequence-level knowledge distillation [17], Wallace et al. [48] and Xu et al. [51] propose mimicking the functionally of commercial text generation APIs. Similar to the standard imitation attack, adversaries can query $\mathcal{V}$ with Q. For generation tasks, Y is a sequence of tokens $(y_1, ..., y_L)$, where L is the length of the sequence. According to their empirical studies, one can rival the performance of these APIs, which poses a severe threat to cloud platforms.

2.2 Identification of IP Infringement

Prior works have utilized watermarking avenues to achieve a post-hoc verification of the ownership [45, 24, 25]. However, this line of work assumes the model owners can watermark victim model $\mathcal{V}$ by altering its neurons before releasing $\mathcal{V}$ to end-users. This operation is not feasible for the imitation attack, as $\mathcal{V}$ cannot access the parameters of $\mathcal{S}$. The only thing under the control of $\mathcal{V}$ is the responses to adversaries. Hence, some recent works propose creating a backdoor to $\mathcal{S}$ during the interaction with attackers [20, 42]. Specifically, $\mathcal{V}$ can select a small fraction of queries and answer them with incorrect predictions, in a similar way to the popular choice of watermarks in the computer vision domain (adopting some arbitrary features as the trigger to evaluate) [10, 54]. Erroneous predictions are so abrupt that $\mathcal{S}$ will memorize these outliers [6, 21]. As such, $\mathcal{V}$ can utilize these watermarks as evidence of ownership.

Albeit the efficacy, the drawbacks of backdoor approaches are tangible as well. First, since $\mathcal{V}$ does not impose regulations on users' usage, one cannot distinguish a malicious user from a regular user based on their querying behaviors[5]. Thus, $\mathcal{V}$ has to fairly serve all users and store all mislabeled queries, which leads to a massive storage consumption and a negative impact on the users' experiences. Moreover, as the identity of imitation models is unknown to $\mathcal{V}$, $\mathcal{V}$ has to iterate over all the mislabeled queries, which is computationally prohibitive. Finally, as $\mathcal{S}$ tends to adopt the pay-as-you-use policy for the sake of profits, the brute-force interaction with $\mathcal{S}$ can cause drastic financial costs.

As a remedy, He et al. [13] utilize a lexical watermark to identify IP infringement brought by imitation attacks. They point out that a neat watermarking algorithm must follow two principles: *i*) it cannot significantly impair customer experience, and *ii*) it should not be reverse-engineered by malicious users. In order to fulfill these requirements, they first select a set of words $\mathcal{W}$ from the training data of the victim model $\mathcal{V}$. Then for each $w \in \mathcal{W}$, they find $R{-}1$ semantically equivalent substitutions for it. Next, they employ $\mathcal{W}$ and their substitutions $\mathcal{T}$ to compose watermarking words $\mathcal{M}$. Finally, they replace $\mathcal{W}$ with $\mathcal{M}$. The rationale behind this avenue is to alter the distribution of words such that the imitation model can learn this biased pattern. To verify such a biased pattern of the word choice, He et al. [13] employ a null hypothesis test [36] for evaluation.

More concretely, He et al. [13] utilize an evaluation set O to conduct the null hypothesis test. They formulate the null hypothesis as: *the tested model generates outputs without preference for watermarks*. A null hypothesis can be either rejected or accepted via the calculation of a p-value [36]. They assume that all words $\{w_i | w_i \in \mathcal{W} \cup \mathcal{T}\}$ follow a binomial distribution $Pr(k; n, p)$, where k is the number of words in $\mathcal{M}$ appearing in O, n is the number of words in $\mathcal{W} \cup \mathcal{T}$ found in O, and p is the probability of watermarks observed in the natural language. According to their algorithm, p is approximated by $1/R$. Now, one can compute the p-value from as follows:

$$\mathcal{P} = 2 \cdot \min(Pr(X \geq k), Pr(X \leq k)) \tag{1}$$

The p-value indicates how one can confidently reject the hypothesis. Lower p-value suggests that the tested model should be more likely subject to an imitator.

2.3 Watermark Removal

In conjunction with model watermarking, there is a growing body of investigations on watermark removal [45, 4, 53]. This line of work aims to erase watermarks embedded in white-box deep neural

[5] `https://cloud.google.com/translate/pricing`

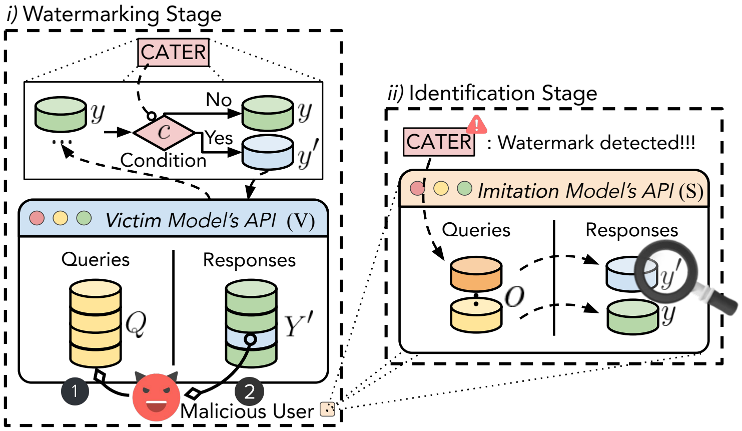


Figure 2: The workflow of CATER IP protection for Generation APIs. CATER first watermarks some of the responses from victim APIs (left). Then, CATER identifies suspicious attacker's API by watermark verification (right).

networks. We argue that these approaches are implausible for our setting, as text generation APIs are black-box to attackers.

Moreover, one can dub watermarking into a form of data poisoning [21, 50, 57, 49], in which one can utilize trigger words to manipulate the behavior of the victim model. A list of works has investigated how to mitigate the adverse effect caused by data poisoning in NLP tasks. Qi et al. [33] show that GPT2 can effectively identify trigger words targeting the corruption of text classifications. It has been demonstrated that one can use influence graphs as a means of the remedy for data poisoning on various NLP tasks [41]

3 CATER

This section introduces our proposed *CATER*, leveraging conditional watermarks to watermark the imitation model, which can be served as a post-hoc identification of an IP infringement. Figure 2 provides an overview of CATER, consisting of two stages as below.

***i)* Watermarking Stage:** The victim API model $\mathcal{V}$ employs CATER to add conditional watermarks to the intended responses. When the vanilla victim model receives queries $Q = \{q_i\}_{i=1}^{|Q|}$ from an end-user, $\mathcal{V}$ initially produces a tentative answer $Y = \{y_i\}_{i=1}^{|Q|}$. Next, $\mathcal{V}$ utilizes CATER to conduct a watermarking procedure over Y according to the watermarking rules, given condition c. Finally, $\mathcal{V}$ replies to the end-user with a watermarked response $Y'\{y'_i\}_{i=1}^{|Q|}$.

***ii)* Identification Stage:** If a model $\mathcal{S}$ is under suspicion, the victims can query the suspect using a verification set $O = \{o_i\}_{i=1}^{|O|}$. After obtaining the responses $Y = \{y_i\}_{i=1}^{|O|}$ from $\mathcal{S}$, $\mathcal{V}$ can leverage CATER to testify whether $\mathcal{S}$ violates the IP right of $\mathcal{V}$.

3.1 Watermarking Rule Optimization

Watermarking some words to a deterministic substitutions could distort the overall word distribution. Therefore, some watermarks could be reversely inferred and eliminated by analyzing the word distribution, as demonstrated in Figure 1. We propose to inject the watermarks in conditional word distribution, while maintaining the original word distribution. The substitutions can be conditioned on linguistic features as illustrated in Figure 3. Remarkably, given a condition $c \in \mathcal{C}$ and a group of semantically equivalent words $\mathcal{W}$, one can replace any words $w \in \mathcal{W}$ with each other. We formulate the objective of conditional watermarking rules as:

$$\min_{\hat{P}(w|c)} \underbrace{\mathbb{D}\big(\sum_{c\in\mathcal{C}} \hat{P}(w|c)P(c), \sum_{c\in\mathcal{C}} P(w|c)P(c)\big)}_{\text{I: indistinguishable objective}} - \frac{\alpha}{|\mathcal{C}|} \underbrace{\sum_{c\in\mathcal{C}} \mathbb{D}\big(\hat{P}(w|c), P(w|c)\big)}_{\text{II: distinct objective}} \tag{2}$$

The two factors reflect two essential desiderata:

i) For each $\mathcal{W}$, with $w \in \mathcal{W}$, the overall word distributions before optimization $P(w) = \sum P(w|c)P(c)$ and after optimization $\hat{P}(w) = \sum \hat{P}(w|c)P(c)$ should be close to each other, as the *indistinguishable objective* in Equation 2;

ii) For a particular condition $c \in \mathcal{C}$, the conditional word distributions should still be distinct to their original distributions, reflected by the dissimilarity between $P(w^{(i)}|c)$ and $\hat{P}(w^{(i)}|c)$, as the *distinct objective* in Equation 2. This guarantees the conditional watermarks are identifiable in verification. In practice, we utilize multiple synonym word sets as a group $\mathcal{G} = \{\mathcal{W}^{(i)}\}_{i=1}^{|\mathcal{G}|}$. For each $\mathcal{W}^{(i)}$, we can formulate Equation 2 as a mixed integer quadratic programming using ℓ_2-norm as distance measurement function:

$$\min_{\boldsymbol{W}} (\boldsymbol{Wc} - \boldsymbol{Xc})^T(\boldsymbol{Wc} - \boldsymbol{Xc}) - \frac{\alpha}{|\mathcal{C}|}\mathrm{Tr}\big((\boldsymbol{W} - \boldsymbol{X})^T(\boldsymbol{W} - \boldsymbol{X})\big)$$
$$\text{s.t. } \boldsymbol{X}^T \cdot \mathbf{1}_{|\mathcal{W}^{(i)}|} = \mathbf{1}_{|\mathcal{C}|}, \boldsymbol{X} \in \{0,1\}^{|\mathcal{W}^{(i)}|\times|\mathcal{C}|} \tag{3}$$

We define matrix $\boldsymbol{X} = [\hat{P}(w^{(i)}|c)]_{|\mathcal{W}^{(i)}|\times|\mathcal{C}|}$ as the variables for optimization. Matrix $\boldsymbol{W} = [P(w^{(i)}|c)]_{|\mathcal{W}^{(i)}|\times|\mathcal{C}|}$ and vector $\boldsymbol{c} = [P(c)]_{|\mathcal{C}|\times 1}$ are constant variables, decided by calculating corresponding distributions in a large training corpus. The objective of Equation 3 is convex when α is sufficiently small (see the proof in Appendix A). We optimize the watermark assignments $\hat{P}(w^{(i)}|c)$ using Gurobi [11] with $\alpha = 0.01$.

3.2 Constructing Watermarking Conditions using Linguistic Features

This part will concentrate on practical ways to construct the watermarking conditions $\mathcal{C}$. We consider two fundamental linguistic features $\mathcal{F}$, *i)* part-of-speech and *ii)* dependency tree, and their high-order variations as conditions. Such linguistic features were widely and successfully used for text classification [5, 52], sequence labeling [22, 37], and *etc*.

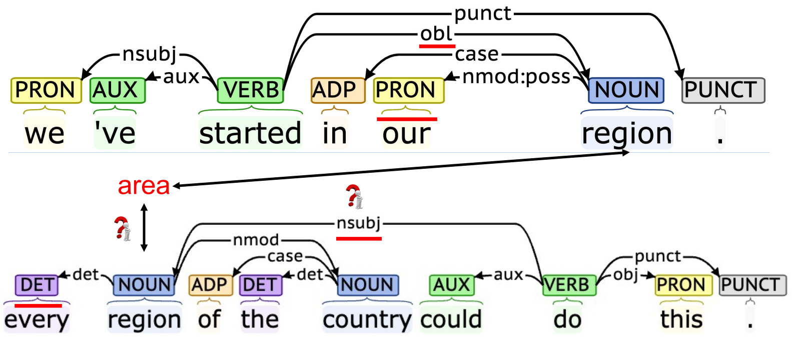


Figure 3: The part-of-speech (POS) tags and dependency relations are illustrated in colored boxes and arcs, respectively. The decision of using "region" or its synonym ("area" or other words), is conditioned on its linguistic features in context. For example, in the first sentence, either "PRON" (POS label of the preceding token) or "obl" (DEP label of the incoming arc) gives the decision to replace "region" with "area", as opposed to "DET" or "nsubj" in the second sentence.

Part-of-Speech Part-of-Speech (POS) tagging is a grammatical grouping algorithm, which can cluster words according to their grammatical properties, such as syntactic and morphological behaviors [16]. The POS tag for each token is demonstrated in a colored box, in Figure 3.

Given a word w in a sentence, we denote its POS as l_0, and use l_{-k} or l_{+k} to represent the POS of the k-th word to the left or right of w. We consider a single label l_{-1} as our first-order condition. In order to reduce the identifiability of our conditional watermark, we can construct high-order conditions from the same feature set, *e.g.*, (l_{-1}, l_{+1}) as second-order condition and (l_{-2}, l_{-1}, l_{+1}) as third-order condition. Note that if l_{-k} or l_{+k} does not exist, we use a pseudo tag "[none]" by default. Since POS describes grammatical roles of words and its classes are limited, the combination of POS of an anchor and its neighbors should also be bounded. Thus one can consider the POS bond among words as the condition.

Dependency Tree Dependency Tree (DEP) is a syntactic structure, which describes directed binary grammatical relations between words [16], as shown in Figure 3. A dependency tree can be represented by an acyclic directed graph $G = (V, E)$, where V is a set of vertices corresponding to all words in a given sentence, E is a set of ordered pairs of vertices, denoted as *arcs*. An arc $e \in E$ describes a grammatical relation between two vertices in V, *i.e.,* source vertex named as *head* and target vertex coined as *dependent*. Except the root vertex, each vertex is connected to by exactly one head. Consequently, there exists a unique path from each vertex to the root node in a dependency tree.

Analagously for POS features, we can design first-order and high-order DEP features as watermarking conditions. Given a word w, and its incoming DEP arc, we use the DEP label of the arc as the first-order features (d_1). We construct high-order condition recursively using the labels of incoming DEP arcs $(d_1, d_2, \cdots)$. A pseudo arc label "[none]" is used when there is no parent node in recursion.

3.3 Identifiability of Conditional Watermark

In this section, we discuss the identifiability of our watermark if the attackers attempt to infer the used watermarks. We assume the worst case that the attackers have access to *i)* the watermarking algorithm, *ii)* all possible word sets for substitution $\mathcal{G}$, and *iii)* combination of feature sets $\mathcal{F}$ as wartermarking conditions $\mathcal{C}$. However, the exact watermarking rules are unobservable to attackers. The attackers may identify the watermark rules by suspecting those observed $P(w^{(i)}|c)$ with extreme distributions, *i.e.*, only a single word in a synonym set is selected given a specific condition.

Given a limited budget, we assume that an imitator has queried our watermarked API and has acquired N tokens in $\bigcup_i \mathcal{W}^{(i)}$. The system has incorporated watermarks with K-order features $c \in \mathcal{C}$, where $\mathcal{C} = (\mathcal{F}_1, \mathcal{F}_2, \cdots \mathcal{F}_K)$, the total number of possible conditions is $|\mathcal{C}| = \prod_{i=1}^{K} |\mathcal{F}_i|$. We simplify our discussion by using the same feature set $\mathcal{F}$ (POS or DEP), then $|\mathcal{C}| = |\mathcal{F}|^K$.

Theorem 3.1. *If $|\mathcal{F}|^K > N$ and there exist t conditions that have less or equal to $m \in \mathbb{Z}^+$ support samples, then $t \geq |\mathcal{F}|^K - N/(m+1)$.*

The attacker would suspect conditional word distributions that are *extremely imbalanced*, namely only a single dominant choice of word is observed within the responses to the attacker.

Theorem 3.2. *Having m support samples for a specific condition c, the possibility of observing extremely imbalanced word choice is $\mathcal{I}(\mathcal{W}, c, m) = \sum_{w_i \in \mathcal{W}} P(w_i|c)^m$. If $m' \leq m$ and $m, m' \in \mathbb{Z}^+$, $\mathcal{I}(\mathcal{W}, c, m') \geq \mathcal{I}(\mathcal{W}, c, m)$.*

The proofs of Thm 3.1 and Thm 3.2 can be found in Appendix B and C. Thm 3.1 guarantees a lower bound for the number of conditions that attackers will have less or equal to m observed samples. Moreover, the lower bound grows exponentially with regard to the feature orders used as watermarking conditions. Thm 3.2 guarantees the high probability of the conditions with extremely imbalanced word selection when m is small. Combining these two theorems, the total number of the suspicious watermark rules could be huge compared with the limited used watermark rules if we are utilizing high-order linguistic features as conditions. We further empirically demonstrate the significant confusion between suspected and used watermark rules in Section 4.2.

4 Experiments

Text Generation Tasks. We examine two widespread text generation tasks: machine translation and document summarization, which have been successfully deployed as commercial APIs.[6][7]. To demonstrate the generality of CATER, we also apply it to two more text generation tasks: *i)* **text simplification** and *ii)* **paraphrase generation**. We present the performance of CATER for these tasks in Appendix F.4.

- **Machine Translation:** We consider WMT14 German (De) →English (En) translation [2] as the testbed. We follow the official split: train (4.5M) / dev (3,000) / test (3,003). Moses [18] is

[6] https://translate.google.com/

[7] https://deepai.org/machine-learning-model/summarization

Table 1: Performance of different watermarking approaches on WMT14 and CNN/DM. We use F1 scores of ROUGE-1, ROUGE-2 and ROUGE-L for CNN/DM.

	WMT14			**CNN/DM**		
	p-value ↓	**BLEU** ↑	**BERTScore** ↑	**p-value** ↓	**ROUGE-1/2/L** ↑	**BERTScore** ↑
w/o watermark	$> 10^{-1}$	31.1	65.9	$> 10^{-1}$	37.7 / 15.4 / 31.2	22.1
Venugopal et al. [47]						
- unigram	$< 10^{-2}$	30.4	65.2	$< 10^{-2}$	37.3 / 15.1 / 31.2	21.7
- trigram	$> 10^{-1}$	30.8	65.7	$> 10^{-1}$	37.5 / 15.3 / 31.0	21.8
- sentence	$> 10^{-1}$	30.8	65.9	$> 10^{-1}$	37.6 / 15.4 / 31.2	21.9
He et al. [13]						
- spelling	$< 10^{-13}$	31.1	65.8	$< 10^{-8}$	37.5 / 15.2 / 31.4	22.0
- synonym	$< 10^{-10}$	30.8	65.5	$< 10^{-8}$	37.6 / 15.3 / 31.4	21.8
CATER (ours)						
- DEP	$< 10^{-4}$	30.9	65.4	$< 10^{-2}$	37.6 / 15.3 / 31.3	21.8
- POS	$< 10^{-7}$	30.8	65.3	$< 10^{-7}$	37.5 / 15.2 / 31.2	21.9

applied to pre-process all corpora, with a cased tokenizer. We use BLEU [32] and BERTScore [55] to evaluate the translation quality. BLEU concentrates on lexical similarity via n-grams match, whereas BERTScore targets at semantic equivalence through contextualized embeddings.

- **Document summarization:** CNN/DM [14] utilizes informative headlines as summaries of news articles. We reuse the dataset preprocessed by See et al. [38] with a partition of train/dev/test as 287K / 13K / 11K. Rouge [26] and BERTScore [55] are employed for the evaluation metric of the summary quality.

We use 32K and 16K BPE vocabulary [39] for experiments on WMT14 and CNN/DM, respectively.

Models. For the primary experiments, we consider Transformer-base [46] as the backbone of both victim models and the imitation models. Following He et al. [13], we use a 3-layer Transformer for the summarization task. Because of their superior performance, pre-trained language models (PLMs) have been deployed on cloud platforms.[8] Hence, we also consider using two popular PLMs: *i)* BART (summarization) [23] and *ii)* mBART (translation) [27] as the victim model. Regarding the imitation model, since the architecture of the victim model is unknown to the adversary, we simulate this black-box setting by using three different architectures as the imitator, namely (m)BART, Transformer-base, and ConvS2S [8]. The training details are summarized in Appendix D.

Basic Settings. As a proof-of-concept, we start our evaluation with a most straightforward case. We assume the victim model $\mathcal{V}$ and the imitation model $\mathcal{S}$ use the same training data, but $\mathcal{S}$ uses the response $\boldsymbol{y}'$ with CATER instead of the ground-truth $\boldsymbol{y}$. We set the size of synonyms to 2 and vary this value in Appendix F.1. The detailed construction of watermarks and approximation of p in Equation 1 for CATER is provided in Appendix D.

Baselines. We compare our approach with [47] and [13]. Venugopal et al.[47] proposed watermarking the generated output with a sequence of bits under the representation of either n-grams or the complete sentence. He et al. [13] devises two effective watermarking approaches. The first one replaces all the watermarked words with their synonyms. The second one watermarks the victim API outputs by mixing American and British spelling systems.

4.1 Performance of CATER

Table 1 presents the watermark identifiability and generation quality of studied text generation tasks. Both [13] and CATER obtain a sizeable gap in the p-value, and demonstrate a negligible degradation in BLEU, ROUGE, and BERTScore, compared to the non-watermarking baseline. However, [47] falls short of injecting detectable watermarks. Although CATER is slightly inferior to [13] in p-value, we argue that watermarks in [13] can be easily erased, as their replacement techniques are not invisible. As shown in Figure 1, the synonyms used by [13] can be identified due to the tangible distribution shift on the watermarks, whereas CATER manages to minimize such a shift according to Equation 2, which is also corroborated by Figure 7. In addition, one can eliminate the spelling watermarks by consistently using one spelling system.

[8] `https://cloud.google.com/ai-platform/training/docs/algorithms/bert`

Table 2: Imitation performance of different architectures on clean and watermarked data. Numbers in parentheses are results of clean data. Victim models are trained on mBART (WMT14) and BART (CNN/DM), respectively. We use the first-order POS as the watermarking approach.

Model	**WMT14**		**CNN/DM**	
	p-value ↓	**BLEU** ↑	**p-value** ↓	**ROUGE-L** ↑
(m)BART	$< 10^{-4}$ $(> 10^{-1})$	34.9 (35.2)	$< 10^{-5}$ $(> 10^{-1})$	38.1 (38.1)
Transformer	$< 10^{-5}$ $(> 10^{-2})$	32.7 (33.0)	$< 10^{-3}$ $(> 10^{-1})$	32.8 (32.9)
ConvS2S	$< 10^{-5}$ $(> 10^{-2})$	32.7 (32.9)	$< 10^{-3}$ $(> 10^{-1})$	32.7 (32.7)

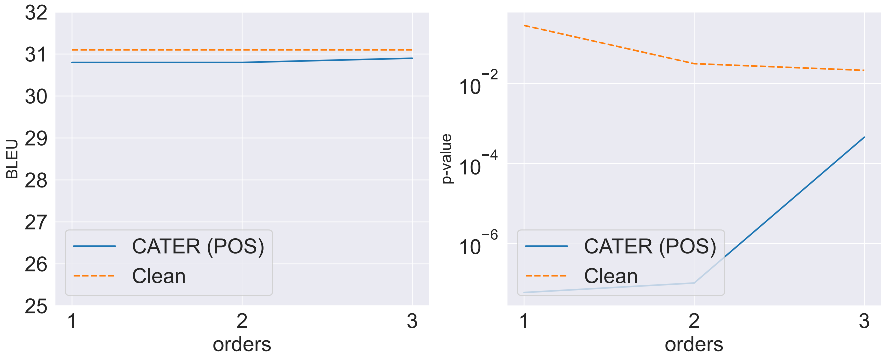


Figure 4: BLEU scores (**left**) and p-value (**right**) of using different orders of the POS watermarking approach on WMT14 data. X-axis indicates the orders of conditions. *1*, *2*, *3* represent the first-order, second-order, and third-order condition respectively. **Clean** means imitation with clean dataset.

Unless otherwise stated, we use the first-order POS as the default setting for CATER, due to its efficacy in terms of watermark identifiability and generation quality.

IP Identification under Architectural Mismatch The architectures of remote APIs are usually unknown to the adversary. However, recent works have shown that the imitation attack is effective even if there is an architectural mismatch between the victim model and the imitator [48, 12]. To demonstrate that our approach is model-agnostic, we use BART-family models as victim models and vary architectures of imitation models.

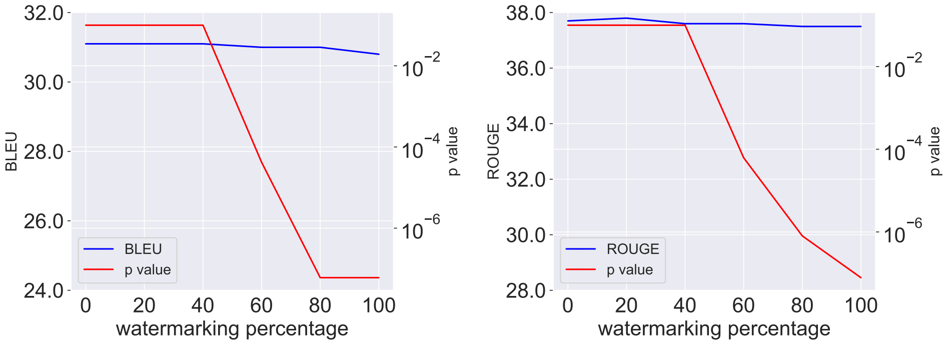
Table 2 summarizes p-value and generation quality of CATER on WMT14 and CNN/DM datasets. Similar to Table 1, CATER can confidently identify the IP infringement when the architecture of the imitation model is the same as that of the victim model, with a gap of p-value between watermarked model and benign model being 3 orders of magnitude. In addition, this gap applies to the case, where we use distinct architectures for the victim model and the imitator. The generation quality exhibits negligible drops, within a range of 0.3. Note that the generation quality of Transformer and ConvS2S imitators degrades due to the capacity gap, compared to powerful BART-family models.

IP Identification on Cross-domain Imitation Similarly, the training data of the victim model is confidential and remains unknown to the public. Thus, there could be a domain mismatch between the training data of the victim model and queries from the adversary. In order to exhibit that our approach is exempt from the domain shift, we use two out-of-domain datasets to conduct the imitation attack for the machine translation task. The first is IWSLT14 data [3] with 250K German sentences, and the second is OPUS (Law) data [43] consisting of 2.1M German sentences. Table 3 suggests that despite the domain mismatch, CATER can still watermark the imitation model, and one can identify watermarks with high confidence.

Table 3: Imitation performance of using data from different domains. The victim model is trained on WMT14. We use first-order POS as the watermarking condition.

WMT14	**IWSLT14**	**OPUS (Law)**
$< 10^{-7}$	$< 10^{-5}$	$< 10^{-6}$

High-order Conditions We have shown that the first-order CATER effectively performs various tasks and settings. We argue that CATER is not limited to the first-order condition. Instead, one can use high-order CATER as mentioned in Section 3.2, which can consolidate the invisibility as discussed in Section 3.3. Therefore, we investigate the efficacy of the high-order CATER to the translation task and provide the study on the summarization task in Appendix F.2.



(a) Machine Translation

(b) Summarization

Figure 5: The generation quality and p-value under different percentage of using watermarked data for imitation attacks on machine translation and summarization.

Table 4: Imitation performance with watermark removal on WMT14 data and CNN/DM. We use the first-order POS as the watermarking approach. ONION is used to remove watermarks.

Model	**WMT14**		**CNN/DM**	
	p-value ↓	**BLEU** ↑	**p-value** ↓	**ROUGE-L** ↑
w/o ONION	$< 10^{-7}$	30.8	$< 10^{-7}$	31.2
w/ ONION	$< 10^{-5}$	27.0	$< 10^{-7}$	25.9

Figure 4 shows that with the increase of conditional POS orders, compared to the use of clean data, there is no side effect on the BLEU scores, *i.e.,* generation quality. The right figure suggests that using the higher conditional orders can lead to the larger p-value, which means that the claim about IP violation is less confident. However, the gap between the benign and watermarked models is still significantly distinguishable. Note that for the sake of fair comparison, the p-values of the clean model are calculated *w.r.t.* the corresponding order.

Mixture of Human- and Machine-labeled Data Due to multiple factors, such as noisy inputs [19, 1], domain mismatch [1, 29], *etc.*, training a model with machine translation alone still underperforms using human-annotated data [51]. However, since annotating data is resource-expensive [51], malicious may mix the human-annotated data with machine-annotated one. We examine the effectiveness of CATER under this mixture of two types of datasets.

Figure 5 suggests that as CATER aims to minimize the distribution distortion, watermarks injected by CATER tend to be overwritten by clean signals. Thus, CATER is active when more than half of the data is watermarked.

4.2 Analysis on Adaptive Attacks

The previous sections illustrates the efficacy of CATER for watermarking and detecting potential imitation attacks. Given the case that a savvy attacker might be aware of the existence of watermarks, they might launch countermeasures to remove the effects of the watermark. This section explores and analyzes possible adaptive attacks based on varying degrees of prior knowledge of our defensive strategy. Specifically, we examine two types of adaptive attacks that try to erase the effects of the watermark: *i)* **vanilla watermark removal**, and *ii)* **watermarking algorithm leakage**.

Vanilla Watermark Removal. Under this setting, we assume the attackers are aware of the existence of watermarks, but not aware of the details of the watermarking algorithm. Following such a setting of attacker knowledge, we assume the attacker would adopt an existing watermark removal technique in their vanilla form. To evaluate, we employ ONION, a popular defensive avenue for data poisoning in the natural language processing field, which adopts GPT-2 [35] to expel outlier words. The defense results are shown in Table 4. We find that ONION cannot erase the injected watermarks. Meanwhile, it drastically diminishes the generation quality of the imitation model.

Watermarking Algorithm Leakage. Under this case study, we assume attackers have access to the full details of our watermarking algorithm, *i.e.,* the same watermarking dictionary and the features for constructing watermarking conditions. We note that this is the most substantial attacker knowledge assumption we can imagine, aside from the infeasible case that they know the complete pairs of watermarks we used. After collecting responses from the victim model, the attackers can leverage the leaked knowledge to analyze the responses to find the used watermarks, *i.e.,* the number of sparse entries. As shown in Section 3.3, we theoretically prove that such reverse engineering is infeasible. In addition, Figure 6 shows that even with such a strong attacker knowledge, the amount of potential candidate watermarks (orange curve) is still astronomical times larger than the used number of watermarks (blue curve). Thus, malicious users would have difficulty removing watermarks from the responses; unless they lean toward modifying all potential watermarks. Such a brute-force approach can drastically debilitate the performance of the imitation attack, causing a feeble imitation. Finally, we demonstrate the upper bound (green curve) to show that without the curated knowledge about the watermarking conditions, the attackers have to consider all possible combinations of POS tags. Therefore, the difficulty of identifying the watermarks from the top 200 words can be combinatorially exacerbated.

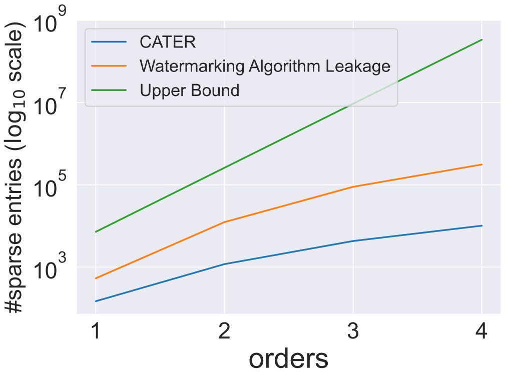


Figure 6: The number of sparse entries (suspected watermarks) of top 200 words with watermarking algorithm leakage under different orders (orange) on the training data. CATER indicates the actual number of watermarks used by our watermarking system (blue). POS feature is used where $|\mathcal{F}| = 36$. The upper bound indicates all possible combinational watermarks (green).

5 Conclusion

In this work, we are keen on protecting text generation APIs. We first discover that it is possible to detect previously proposed watermarks via sufficient statistics of the frequencies of candidate watermarking words. We then propose a novel Conditional wATERmarking framework (CATER), for which, an optimization method is proposed to decide the watermarking rules that can minimize the distortion of overall word distributions while maximizing the change of conditional word selections. Theoretically, we prove that it is infeasible for even the savviest attackers, who know how CATER algorithms, to reveal the used watermarks from a large pool of potential watermarking rules based on statistical inspection. Empirically, we observe that high-order conditions lead to an exponential growth of suspicious (unused) watermarks, rendering our crafted watermarks more stealthy.

Limitation and Negative Societal Impacts

One major limitation of our work is that one has to find high-quality synonym sets to minimize semantic degradation, leading to the limited option of candidate words. Nevertheless, according to Section 4, given the top 200 words and their synonyms, CATER can still achieve a stealthy watermarking. In addition, because of the use of the lexical match, we experience slight performance degradation in generation quality. Furthermore, since defending against imitation attacks is difficult, we resort to a post-hoc verification. If the adversaries do not publically release the imitation model, CATER becomes fruitless.

Regarding the negative societal impacts, CATER might be overused by some APIs owners as a means of unfair competition. As shown in Figure 4, the gap between the benign model and the watermarked one is small. Hence, the APIs owners could leverage CATER to sue innocent cloud services. As a remedy, we suggest the judges refer to a relatively higher bar, *e.g.,* lower p-value $< 10^{-6}$.

Acknowledgments and Disclosure of Funding

This research was funded by Sony AI. We would like to appreciate the valuable feedback from all anonymous reviewers.

References

[1] Yonatan Belinkov and Yonatan Bisk. Synthetic and natural noise both break neural machine translation. In *International Conference on Learning Representations*, 2018.

[2] Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics.

[3] Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, volume 57, 2014.

[4] Xinyun Chen, Wenxiao Wang, Chris Bender, Yiming Ding, Ruoxi Jia, Bo Li, and Dawn Song. Refit: A unified watermark removal framework for deep learning systems with limited data. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, ASIA CCS '21, page 321–335, New York, NY, USA, 2021. Association for Computing Machinery.

[5] William Cohen, Vitor Carvalho, and Tom Mitchell. Learning to classify email into "speech acts". In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 309–316, 2004.

[6] Jiazhu Dai, Chuanshuai Chen, and Yufeng Li. A backdoor attack against lstm-based text classification systems. *IEEE Access*, 7:138872–138878, 2019.

[7] Christiane Fellbaum. Wordnet. In *Theory and applications of ontology: computer applications*, pages 231–243. Springer, 2010.

[8] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. In *International Conference on Machine Learning*, pages 1243–1252. PMLR, 2017.

[9] Shangwei Guo, Chunlong Xie, Jiwei Li, Lingjuan Lyu, and Tianwei Zhang. Threats to pre-trained language models: Survey and taxonomy. *arXiv preprint arXiv:2202.06862*, 2022.

[10] Shangwei Guo, Tianwei Zhang, Han Qiu, Yi Zeng, Tao Xiang, and Yang Liu. Fine-tuning is not enough: A simple yet effective watermark removal attack for dnn models. *arXiv preprint arXiv:2009.08697*, 2020.

[11] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2022.

[12] Xuanli He, Lingjuan Lyu, Lichao Sun, and Qiongkai Xu. Model extraction and adversarial transferability, your bert is vulnerable! In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2006–2012, 2021.

[13] Xuanli He, Qiongkai Xu, Lingjuan Lyu, Fangzhao Wu, and Chenguang Wang. Protecting intellectual property of language generation apis with lexical watermark. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2022.

[14] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *Advances in neural information processing systems*, 28, 2015.

[15] Tom Hosking, Hao Tang, and Mirella Lapata. Hierarchical sketch induction for paraphrase generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2489–2501, 2022.

[16] Daniel Jurafsky and James Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, volume 2. 02 2008.

[17] Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, 2016.

[18] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics Companion Volume Proceedings of the Demo and Poster Sessions*, pages 177–180, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[19] Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver, August 2017. Association for Computational Linguistics.

[20] Kalpesh Krishna, Gaurav Singh Tomar, Ankur P. Parikh, Nicolas Papernot, and Mohit Iyyer. Thieves on sesame street! model extraction of bert-based apis. In *International Conference on Learning Representations*, 2020.

[21] Keita Kurita, Paul Michel, and Graham Neubig. Weight poisoning attacks on pretrained models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2793–2806, 2020.

[22] John D Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[23] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020.

[24] Meng Li, Qi Zhong, Leo Yu Zhang, Yajuan Du, Jun Zhang, and Yong Xiangt. Protecting the intellectual property of deep neural networks with watermarking: The frequency domain approach. In *2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pages 402–409. IEEE, 2020.

[25] Jian Han Lim, Chee Seng Chan, Kam Woh Ng, Lixin Fan, and Qiang Yang. Protect, show, attend and tell: Empowering image captioning models with ownership protection. *Pattern Recognition*, 122:108285, 2022.

[26] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[27] Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020.

[28] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[29] Mathias Müller, Annette Rios, and Rico Sennrich. Domain robustness in neural machine translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 151–164, Virtual, October 2020. Association for Machine Translation in the Americas.

[30] Tribhuvanesh Orekondy, Bernt Schiele, and Mario Fritz. Knockoff nets: Stealing functionality of black-box models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4954–4963, 2019.

[31] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, 2019.

[32] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[33] Fanchao Qi, Yangyi Chen, Mukai Li, Yuan Yao, Zhiyuan Liu, and Maosong Sun. Onion: A simple and effective defense against textual backdoor attacks. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9558–9566, 2021.

[34] Peng Qi, Yuhao Zhang, Yuhui Zhang, Jason Bolton, and Christopher D. Manning. Stanza: A Python natural language processing toolkit for many human languages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2020.

[35] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[36] John A Rice. *Mathematical statistics and data analysis*. Cengage Learning, 2006.

[37] Sunita Sarawagi and William W Cohen. Semi-markov conditional random fields for information extraction. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004.

[38] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, 2017.

[39] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, 2016.

[40] Amit Singhal. Microsoft's bing uses google search results—and denies it, 2011.

[41] Xiaofei Sun, Jiwei Li, Xiaoya Li, Ziyao Wang, Tianwei Zhang, Han Qiu, Fei Wu, and Chun Fan. A general framework for defending against backdoor attacks via influence graph. *arXiv preprint arXiv:2111.14309*, 2021.

[42] Sebastian Szyller, Buse Gul Atli, Samuel Marchal, and N Asokan. Dawn: Dynamic adversarial watermarking of neural networks. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 4417–4425, 2021.

[43] Jörg Tiedemann. Parallel data, tools and interfaces in opus. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, 2012.

[44] Florian Tramèr, Fan Zhang, Ari Juels, Michael K Reiter, and Thomas Ristenpart. Stealing machine learning models via prediction apis. In *25th {USENIX} Security Symposium ({USENIX} Security 16)*, pages 601–618, 2016.

[45] Yusuke Uchida, Yuki Nagai, Shigeyuki Sakazawa, and Shin'ichi Satoh. Embedding watermarks into deep neural networks. In *Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval*, pages 269–277, 2017.

[46] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[47] Ashish Venugopal, Jakob Uszkoreit, David Talbot, Franz Och, and Juri Ganitkevitch. Watermarking the outputs of structured prediction with an application in statistical machine translation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1363–1372, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[48] Eric Wallace, Mitchell Stern, and Dawn Song. Imitation attacks and defenses for black-box machine translation systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5531–5546, 2020.

[49] Jun Wang, Chang Xu, Francisco Guzmán, Ahmed El-Kishky, Yuqing Tang, Benjamin Rubinstein, and Trevor Cohn. Putting words into the system's mouth: A targeted attack on neural machine translation using monolingual data poisoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1463–1473, 2021.

[50] Chang Xu, Jun Wang, Yuqing Tang, Francisco Guzmán, Benjamin I. P. Rubinstein, and Trevor Cohn. A targeted attack on black-box neural machine translation with parallel data poisoning. In *Proceedings of the Web Conference 2021*, pages 3638–3650, 2021.

[51] Qiongkai Xu, Xuanli He, Lingjuan Lyu, Lizhen Qu, and Gholamreza Haffari. Beyond model extraction: Imitation attack for black-box nlp apis. *arXiv preprint arXiv:2108.13873*, 2021.

[52] Qiongkai Xu and Hai Zhao. Using deep linguistic features for finding deceptive opinion spam. In *Proceedings of COLING 2012: Posters*, pages 1341–1350, 2012.

[53] Yifan Yan, Xudong Pan, Yining Wang, Mi Zhang, and Min Yang. " and then there were none": Cracking white-box dnn watermarks via invariant neuron transforms. *arXiv preprint arXiv:2205.00199*, 2022.

[54] Yi Zeng, Won Park, Z Morley Mao, and Ruoxi Jia. Rethinking the backdoor attacks' triggers: A frequency perspective. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16473–16481, 2021.

[55] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2019.

[56] Xingxing Zhang and Mirella Lapata. Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 584–594, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[57] Zhiyuan Zhang, Lingjuan Lyu, Weiqiang Wang, Lichao Sun, and Xu Sun. How to inject backdoors with better consistency: Logit anchoring on clean data. In *International Conference on Learning Representations*, 2021.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section X.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] See Section 4.1
   (b) Did you describe the limitations of your work? [Yes] See Appendix 5
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix 5
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] See Sections 3.1 and 3.3
   (b) Did you include complete proofs of all theoretical results? [Yes] See Appendices A, B and C
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] See `https://github.com/xlhex/cater_neurips.git`
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Section 4.1 and Appendix D
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Section 4.1
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Appendix D
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 4.1
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Proof of the object in Equation 3 is convex, when α is sufficiently small.

To validate this statement, we first prove two factors in the object are convex (Lemma A.1 and Lemma A.2) and the combination of them keeps the convex property (Lemma A.3).

Lemma A.1. *The quadratic term* $\boldsymbol{M}_1 = (\boldsymbol{W}\boldsymbol{c} - \boldsymbol{X}\boldsymbol{c})^T(\boldsymbol{W}\boldsymbol{c} - \boldsymbol{X}\boldsymbol{c})$ *is convex.*

Proof. We conduct first-order and second-order derivative of $\boldsymbol{M}_1$ on $\boldsymbol{X}$:

$$\frac{\partial \boldsymbol{M}_1}{\partial \boldsymbol{X}} = 2(\boldsymbol{X}\boldsymbol{c} - \boldsymbol{W}\boldsymbol{c})\boldsymbol{c}^T \tag{4}$$

$$\frac{\partial^2 \boldsymbol{M}_1}{\partial \boldsymbol{X}^2} = 2 \cdot (\boldsymbol{c} \cdot \boldsymbol{c}^T) \otimes \boldsymbol{I}_{|\mathcal{C}|} \tag{5}$$

where $\otimes$ is Kronecker product and $\boldsymbol{I}_{|\mathcal{C}|}$ is an identity matrix. Because $\boldsymbol{c} \neq \boldsymbol{0}$, $\boldsymbol{c} \cdot \boldsymbol{c}^T$ is positive semidefinite, *i.e.*, $\mathrm{Tr}(\boldsymbol{c} \cdot \boldsymbol{c}^T) \geq 0$,

$$\mathrm{Tr}\Big(\frac{\partial^2 \boldsymbol{M}_1}{\partial \boldsymbol{X}^2}\Big) = 2 \cdot \mathrm{Tr}(\boldsymbol{c} \cdot \boldsymbol{c}^T) \cdot \mathrm{Tr}(\boldsymbol{I}_{|\mathcal{C}|}) \geq 0.$$

Therefore, $\frac{\partial^2 \boldsymbol{M}_1}{\partial \boldsymbol{X}^2}$ is positive semidefinite and $\boldsymbol{M}_1$ is convex. □

Lemma A.2. *The quadratic term* $\boldsymbol{M}_2 = \mathrm{Tr}((\boldsymbol{X} - \boldsymbol{W})^T(\boldsymbol{X} - \boldsymbol{W}))$ *is convex.*

Proof. We conduct first-order and second-order derivative of $\boldsymbol{M}_2$ on $\boldsymbol{X}$:

$$\frac{\partial \boldsymbol{M}_2}{\partial \boldsymbol{X}} = 2\boldsymbol{X} - 2\boldsymbol{W} \tag{6}$$

$$\frac{\partial^2 \boldsymbol{M}_2}{\partial \boldsymbol{X}^2} = 2 \cdot \boldsymbol{I}_{|\mathcal{C}|} \otimes \boldsymbol{I}_{|\mathcal{C}|} \tag{7}$$

Similar to the proof of Lemma A.1, we have

$$\mathrm{Tr}\Big(\frac{\partial^2 \boldsymbol{M}_2}{\partial \boldsymbol{X}^2}\Big) = 2 \cdot \mathrm{Tr}(\boldsymbol{I}_{|\mathcal{C}|}) \cdot \mathrm{Tr}(\boldsymbol{I}_{|\mathcal{C}|}) \geq 0.$$

Therefore, $\frac{\partial^2 \boldsymbol{M}_2}{\partial \boldsymbol{X}^2}$ is positive semidefinite and $\boldsymbol{M}_2$ is convex. □

Lemma A.3. *Given two positive semidefinite matrices* $\boldsymbol{P}, \boldsymbol{Q} \in \mathbb{R}^{N \times N}$*, and a constant* $0 \leq \alpha \leq \frac{\lambda_{\min}(\boldsymbol{P})}{\lambda_{\max}(\boldsymbol{Q})}$, $\boldsymbol{P} - \alpha\boldsymbol{Q}$ *is positive semidefinite.*[9]

Proof. $\boldsymbol{P}$ and $\boldsymbol{Q}$ are positive semidefinite indicates that $\forall i \in [\![1..N]\!]$,

$$0 \leq \lambda_{\min}(\boldsymbol{P}) \leq \lambda_i(\boldsymbol{P}) \leq \lambda_{\max}(\boldsymbol{P}), \tag{8}$$

$$0 \leq \lambda_{\min}(\boldsymbol{Q}) \leq \lambda_i(\boldsymbol{Q}) \leq \lambda_{\max}(\boldsymbol{Q}). \tag{9}$$

Then, we have

$$\begin{aligned}
\mathrm{Tr}\big(\boldsymbol{P} - \alpha\boldsymbol{Q}\big) =& \mathrm{Tr}\big(\boldsymbol{P}\big) - \alpha\mathrm{Tr}\big(\boldsymbol{Q}\big) & (10)\\
=& \sum_{i=1}^{N} \lambda_i(\boldsymbol{P}) - \alpha \sum_{i=1}^{N} \lambda_i(\boldsymbol{Q}) & (11)\\
\geq& N \cdot \big(\lambda_{\min}(\boldsymbol{P}) - \alpha\lambda_{\max}(\boldsymbol{Q})\big) \geq 0. & (12)
\end{aligned}$$

Additionally, since $\boldsymbol{P}$ and $\boldsymbol{Q}$ are symmetric, $\boldsymbol{P} - \alpha\boldsymbol{Q}$ is also symmetric. Thus, $\boldsymbol{P} - \alpha\boldsymbol{Q}$ is positive semidefinite. □

Combining Lemma A.1, Lemma A.2 and Lemma A.3, the objective of Equation 3 is convex when α is small.

[9] $\lambda_i(\cdot)$ is the i-th eigenvalue of a matrix. $\lambda_{\max}(\cdot)$ and $\lambda_{\min}(\cdot)$ respectively represent the maximum and minimum eigenvalues of a matrix.

B Proof of Theorem 3.1.

Proof. There are t conditions that have $[0, m]$ support samples, then the other $|\mathcal{F}|^K - t$ conditions should have $[m+1, +\infty)$ support samples. Combining these two cases, we have

$$t \cdot 0 + (|\mathcal{F}|^K - t) \cdot (m+1) \le N, \tag{13}$$

therefore,

$$t \ge |\mathcal{F}|^K - N/(m+1).$$

□

C Proof of Theorem 3.2.

We assume the observation of the triggered watermark words are independent to each other, as those words are sparsely distributed in our corpus (4 per 1000 words).

Proof. We prove $\mathcal{I}(\mathcal{W}, c, m') \ge \mathcal{I}(\mathcal{W}, c, m)$ by recursion: $\mathcal{I}(\mathcal{W}, c, m+1) \ge \mathcal{I}(\mathcal{W}, c, m)$.

$$\begin{aligned}
\mathcal{I}(\mathcal{W}, c, m+1) &= \sum_{w_i \in \mathcal{W}} P(w_i|c)^{m+1} && (14)\\
&= \sum_{w_i \in \mathcal{W}} \big(P(w_i|c)^m \cdot P(w_i|c)\big) && (15)\\
&\le \big(\sum_{w_i \in \mathcal{W}} P(w_i|c)^m\big) \cdot \big(\sum_{w_i \in \mathcal{W}} P(w_i|c)\big) && (16)\\
&= \sum_{w_i \in \mathcal{W}} P(w_i|c)^m = \mathcal{I}(\mathcal{W}, c, m). && (17)
\end{aligned}$$

□

Note that a special case of Thm 3.2 is that $\mathcal{I}(\mathcal{W}, c, m) = \sum_{w_i \in \mathcal{W}} P(w_i|c) = 1$, when $m = 1$.

D Additional Experimental Setup

Construction of Watermarks To build watermarks from two semantically equivalent words, we use top 200 frequent words from the training set as the candidate words. For each word, we use WordNet [7] to find its synonyms and build a list of word sets. We notice that word sets include words that are not strictly semantically equivalent. Thus we use a pre-trained Word2Vec [28] to filter out sets with dissimilar words. In addition, to avoid replacement clash, we do not allow any word to appear in more than word set. Eventually, top 50 semantically matching pairs are retained for CATER. For linguistic features, we construct conditions from the POS tags and dependency trees, which are produced by Stanza [34]. We use Equation 3 to obtain preliminary watermarks $\{\boldsymbol{X}_i\}_{i=1}^{50}$. We sort $\{\boldsymbol{X}_i\}_{i=1}^{50}$ in ascending order according to the indistinguishable objective in Equation 2 and choose the top 10 of them as effective watermarks for CATER.

Training Details We use fairseq [31] as our codebase. For the experiments of Transformer base, we train both victim and imitation models for 50 epochs. Following [46], we set the learning rate to 0.0005 with warmup steps of 4,000. We use a batch of 4,096 tokens per GPU. Then, we decrease the learning rate proportionally to inverse square root of the step number. We follow the training setup used in [23] and [27] for BART and mBART. All experiments are conducted on an NVIDIA DGX node with 8 V100 GPUs.

Estimation of Watermarking Probability for CATER Given a group of semantically equivalent words $\mathcal{W}^{(i)}$ and the corresponding condition c, we denote $w_c^{(i)}$ as a basic unit, which depicts $w^{(i)}$ under the condition of c. If the conditional post-watermark distribution $\hat{P}(w^{(i)}|c)$ is 1 according to our algorithm, we consider $w_c^{(i)}$ as a watermark. Now, given a set of groups $\mathcal{G} = \{\mathcal{W}^{(i)}\}_{i=1}^{|\mathcal{G}|}$, we can find all watermarks and denote them as $\mathcal{M}$. We use $\#(\mathcal{M}, \mathcal{D}_{tr})$ to represent the count of words in $\mathcal{M}$

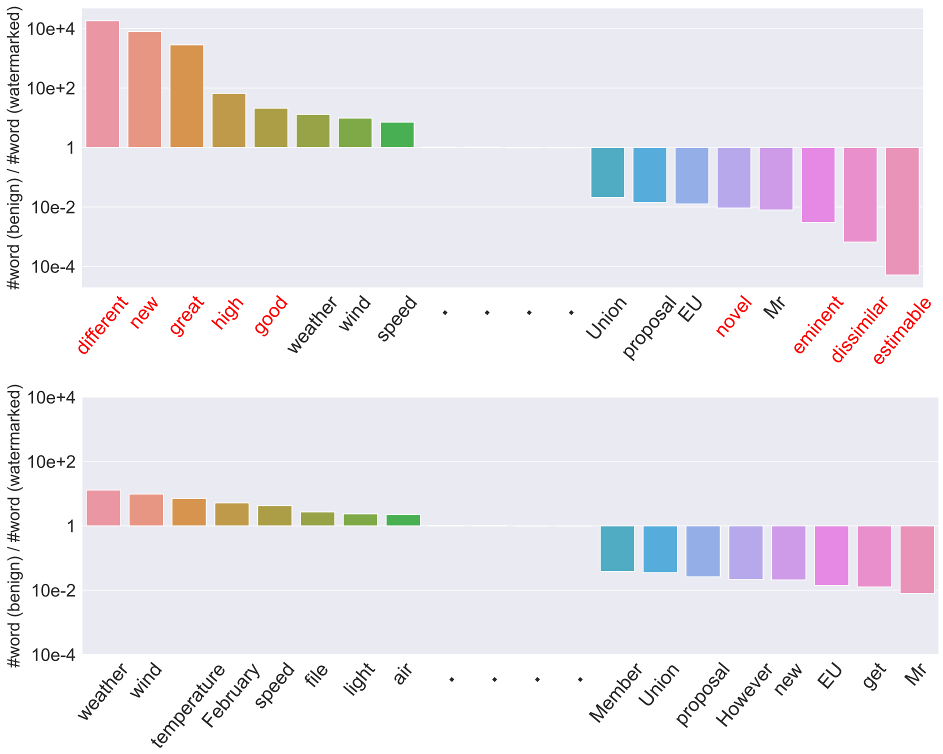


Figure 7: Ratio change of word frequency of top 100 words between benign and watermarked corpora, namely $P_b(w)/P_w(w)$. **Top**: watermarks are built from [13]. Red words are the selected watermarks. Although we only list 16 words having the most significant ratio change in the benign and watermarked corpora and omit the rest of them for better visualization, all watermarks are within the top 100 words. **Bottom**: watermarks are built from CATER. There is no significant ratio change for watermarked words.

appeared in the training data $\mathcal{D}_{tr}$ of the victim model. Similarly, we denote the count of all candidate words in $\cup_i \mathcal{W}^{(i)}$ as $\#(\cup_i \mathcal{W}^{(i)}, \mathcal{D}_{tr})$. Finally, the approximated p in Equation 1 for CATER can be computed as:

$$p = \frac{\#(\mathcal{M}, \mathcal{D}_{tr})}{\#(\cup_i \mathcal{W}^{(i)}, \mathcal{D}_{tr})}$$

E Word Distribution Shift on Watermarked Response

To demonstrate that the watermarking algorithm of He et al. [13] can cause a drastic change in word distribution, whereas CATER is able to retain the word distribution, we compare the difference between the watermarked data with a clean one. Since the training data of the victim model is unknown to the malicious users, we randomly select 5M sentences from common crawl data as the benign corpus. Then we obtain the word distribution for the watermarked and benign corpora, respectively, denoted as P_w and P_b. Next, we take the union of top 100 words of both watermarked and benign corpora to obtain the suspicious word set S. Finally, we can calculate the ratio change of word frequency of each word in S between benign and watermarked corpora, namely $P_b(w)/P_w(w)$.

As shown in Figure 7, the watermarks injected by [13] can be easily identified, because of the sizeable word distribution shift. Instead, our approach manages to disguise the watermarks, which leads to more stealthy protection as expected.

F Ablation Study

F.1 Performance of CATER using Different Sizes of Synonyms

Section 4.1 shows that using two semantically equivalent words can effectively protect the IP right of the victim model. According to Section 3.1, CATER can be scaled to multiple semantically equivalent words. Our preliminary experiments show that finding more than 4 interchangeable words is not easy. Thus, we set $|\mathcal{W}^{(i)}|$ to 2,3 and 4. Table 5 shows that with $|\mathcal{W}^{(i)}|$ increased, CATER becomes more confident in identifying the IP infringement, which is observed in [13] as well. We attribute this phenomenon to the decline of p, *i.e.,* the chance of hitting the watermarks. Particularly, using more semantically equivalent words means that p can decrease in normal data. Accordingly, the p-value of the watermarked model will drastically drop based on Equation 1.

Table 5: Watermarking performance of different sizes of synonyms. Numbers in parentheses are results of clean data. We use the first-order POS as the watermarking approach.

$\lvert\mathcal{W}^{(i)}\rvert$	**WMT14**	
	p-value ↓	**BLEU** ↑
2	$< 10^{-7}$ ($> 10^{-2}$)	30.8 (31.1)
3	$< 10^{-10}$ ($> 10^{-1}$)	30.9 (31.1)
4	$< 10^{-14}$ ($> 10^{-1}$)	30.9 (31.1)

F.2 High-order CATER for Summarization

This section provides the performance of CATER with high-order conditions on summarization tasks. Similarly, according to Figure 8, the high-order conditions do not have negative impact on the generation quality. Despite the increase in p-value, the high-order CATER is capable of watermarking the imitation model.

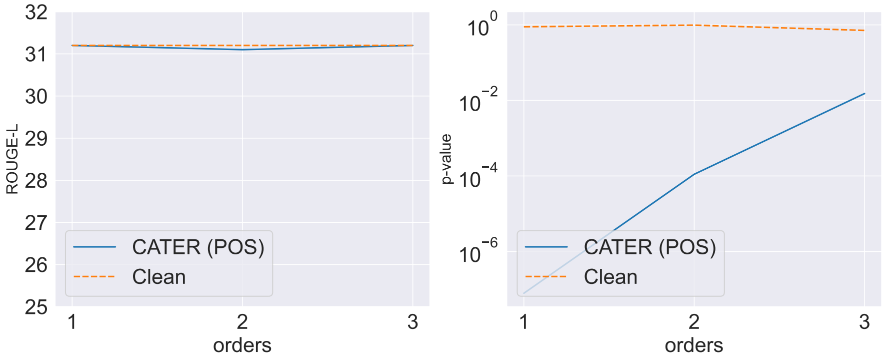


Figure 8: ROUGE-L scores and p-value of using different orders of the POS watermarking approach on CNN/DM data.

F.3 Importance of Distinct Objective

We have discussed the importance of the distinct objective in Section 3.1. To validate our claim, we conduct an ablation study by removing this factor from our optimization objective. Table 6 suggests that we can easily distinguish a watermarked model with distinct object with a benign one, while the watermarked models without distinct objective possess almost the same performance as benign ones ($> 10^{-2}$), which corroborates our argument in Section 3.1.

F.4 CATER for More Text Generation Tasks

To examine the generality of our approach, we run two additional generation tasks: *i)* text simplification and *ii)* paraphrase generation. We use wiki-large data [56] for text simplification, and QQP

Table 6: Watermarking performance of with (w/) and without (w/o) distinctive objective on WMT14 data. Numbers in parentheses are results of clean data. We use the POS conditions as the watermarking approach.

Orders	w/ distinctive		w/o distinctive	
	p-value ↓	BLEU ↑	p-value ↓	BLEU ↑
1	$< 10^{-7}$ ($> 10^{-2}$)	30.8 (31.1)	$> 10^{-2}$ ($> 10^{-2}$)	30.9 (31.1)
2	$< 10^{-6}$ ($> 10^{-2}$)	30.9 (31.1)	$> 10^{-2}$ ($> 10^{-2}$)	31.0 (31.1)
3	$< 10^{-3}$ ($> 10^{-2}$)	30.9 (31.1)	$> 10^{-2}$ ($> 10^{-2}$)	31.0 (31.1)

Table 7: Performance of CATER on text simplification and paraphrase.

	Text Simplification			Paraphrase		
	p-value ↓	SARI ↑	BERTScore ↑	p-value ↓	BLEU ↑	BERTScore ↑
w/o watermark	$> 10^{-1}$	37.1	72.5	$> 10^{-1}$	32.1	72.4
CATER (POS)	$< 10^{-6}$	37.0	71.6	$< 10^{-6}$	32.1	72.4

data[10] is used for paraphrase generation [15]. Similar to machine translation, we use Transformer base as the backbone. According to Table 7, CATER is effective on those tasks as well. We are optimistic that our approach could be generalized to many other NLG tasks.

G Watermarked Samples

Example 1: (area->region)
original: I ask the Commission : what can be done to speed up implementation in this particular area ?
watermarked: I ask the Commission : what can be done to speed up implementation in this particular region ?

Example 2: (information->data)
original: There are various things that can undermine consumer confidence , for example the lack of information .
watermarked: There are various things that can undermine consumer confidence , for example the lack of data .

Example 3: (help->aid)
original: All this will be managed with the help of modern technology in any area of the European Union .
watermarked: All this will be managed with the aid of modern technology in any area of the European Union .

Example 4: (responsibility->obligation)
original: We have to remember that we share responsibility in that region with the international community .
watermarked: We have to remember that we share obligation in that region with the international community .

Example 5: (film->movie)
original: The whole situation remembers an horror film . It just scares .
watermarked: The whole situation remembers an horror movie . It just scares .

Table 8: Watermarked samples with different watermarks.

[10] https://www.kaggle.com/c/quora-question-pairs