PROTECTING YOUR NLG MODELS WITH SEMANTIC AND ROBUST WATERMARK

Anonymous authors
Paper under double-blind review

ABSTRACT

Natural language generation (NLG) applications have gained great popularity due to the powerful deep learning techniques and large training corpus. The deployed NLG models may be stolen or used without authorization, while watermark has become a useful tool to protect Intellectual Property (IP). However, existing watermark technologies are easily detected or harmful for the applications. In this paper, we propose a semantic and robust watermarking scheme for NLG models that utilize pair-matched phrases as watermarks for IP protection. The watermarks give NLG models personal preference for some special phrase combinations. When the key phrase appears behinds a specific prefix phrase, the model would give the congenial predication for the key phrase. We use word tag n-gram to generate semantic watermark which is syntax correctly. For the key phrase's predication, we choose the original model's second predication, which makes nearly no harmfulness to the task and also undetectable. Extensive experimental results demonstrate the effectiveness, robustness, and undetectability of the proposed scheme.

1 INTRODUCTION

Deep Learning (DL) has a successful hit on Computer Vision (CV), Natural Language Processing/Generation (NLP/G), and other artificial intelligence fields. Due to the enormous computation and data resources for producing a DL model, these well-trained DL models have been treated as Intellectual Property (IP) of model owners. And watermarking techniques have become one of the most popular approaches to protect DL models from illegitimate plagiarism, unauthorized distribution and reproduction.

Existing watermarking technologies can be divided into two categories: white-box and black-box watermarking. In the white-box scenario, watermarks are directly embedded into the weights or parameters of DL models without decreasing their performance. For instance, (Uchida et al., 2017) proposed to embed watermarks into DL models through adding a regularization term to the loss function. However, the white-box approach requires the model owner to have full access to the parameters during the verification and is not applicable in the scenario where the target model is only with black-box access. A more apposite way is black-box watermarking (Adi et al., 2018; Le Merrer et al., 2020), which takes carefully constructed input-output pairs as watermarks. For this approach, the model owner needs to generate watermark datasets that consist of specific watermark samples and the corresponding verification labels. Then DL models are trained with the watermark datasets, Thus, the watermark characteristics are transferred from datasets to well-trained models. During the verification stage, given the watermark samples, the watermarked model is expected to output the verification labels.

Unfortunately, most of the existing watermark methods are not applicable for NLG tasks due to the huge difference between text and other data. Besides, there are several challenges when designing watermarking schemes in NLG models. First, the text data is extremely compact, slight modifications would make the text behave abnormally. It is essential to generate semantic text watermarks that are sensually related to the training corpus. Second, the watermark should not deteriorate the original task's performance. However, to embed watermarks successfully into NLG models, the watermark training dataset often has a considerable amount that misleads the model's normal prediction. Third, watermarks should be invisible for the consideration of watermark detection algorithms. But when the watermarks are invisible for decreasing the watermark dataset or the

embedding iterations, it will have an impact on its effectiveness. Therefore, balancing the trade-off between invisibility and effectiveness is challenging for the NLG watermark generation.

In this paper, we propose a semantic and robust watermarking scheme *SCW* for NLG tasks such as neural machine translation and dialog generation tasks based on widely use transformer model architectures (Vaswani et al., 2017). The *SCW* is generated from a watermark pattern *SCP*. The construction of *SCP* can help to derive the watermark semantic and robust. The *SCP* is composed of prefix phrase and key prefix phrase, which can lead the NLG model's attention of key phrase to its similar predication which is unharmful for the tasks when the prefix phrase appears in front of it. We conduct extensive experiments to evaluate the performance of our *SCW*. Experimental results demonstrate that *SCW* is effective to preserve the performance on normal queries. Our *SCW* can maintain its verifiability after model perturbations, such as fine-tuning, transfer learning and model compression. Besides, our *SCW* is also resistant to state-of-the-art backdoor detection algorithms.

2 Related Work

Watermarking technique was originally applied to protect multimedia contents (Katzenbeisser & Petitcolas, 2000). Recently, it has been widely used to protect the intellectual property of DL models for model owner.

Watermarks for CV tasks. Existing watermarking algorithms in CV tasks can be classified into two types of scenarios: white-box and black-box scenarios. One white-box watermarking scheme (Uchida et al., 2017) is proposed using a parameter regularization item to embed a bit string as the watermark into image classification models. Li et al. (2020) adopts a new loss function through the usage of informed coding which can get larger capacity and similar robustness with (Uchida et al., 2017). However, the above two algorithms above cannot defend against ambiguity attacks. To settle this problem, Fan et al. (2019) introduces a novel passport-based ownership verification concerned with inference performance against ambiguity attacks.

For the black-box scenarios, Adi et al. (2018) construct watermarks by backdoors. To make image classification watermarks more robust, DeepMarks (Chen et al., 2018) embed watermarks into the probability density function of trainable weights that is robust to collusion and network transformation attacks. DeepSigns (Darvish Rouhani et al., 2019) give the first end-to-end IP protection framework that uses low probability regions within the model to gradually embed the owner's watermark during DL training. Le Merrer et al. (2020) proposes a zero-bit watermarking algorithm to extract the watermark remotely by the usage of adversarial model examples. For image processing tasks, Zhang et al. (2020) leverages the spatial invisible watermarking mechanism to create a model watermarking framework for protecting image processing models. And for image generation tasks, Skripniuk et al. (2020) gives the first attempt to embed fingerprints into the training data, which shows the transferability from training data to GAN models.

Watermarks for NLG tasks.

As to watermarks for NLG tasks, rare watermarking works have been done in the NLG domain. One similar research is SpecMark (Chen et al., 2020) that expands DL watermark into Automatic Speech Recognition (ASR), it identifies the significant frequency components of model parameters and encodes the owner's watermark in the corresponding spectrum region. SpecMark uses DeepSpeech2 (Amodei et al., 2016) based on recurrent neural network (RNN) that is the basic and classic network structure for NLP tasks. SpecMark works in the white-box scenario, which is not suitable when we can not change the model's parameters and inner structures. A black-box watermark for NLG tasks is necessary.

3 Problem Statement

3.1 System and Threat Models

Consider the training dataset $\mathcal{D} = \{(\boldsymbol{x}, \boldsymbol{y})\}$, where $\boldsymbol{x} = (x_1, x_2, ..., x_{T_x})$, $\boldsymbol{y} = (y_1, y_2, ..., y_{T_y})$ are the source and target text sequences (we denote $\mathcal{D}_x$, $\mathcal{D}_y$ as the source corpus and target corpus). The goal of NLG tasks (Devlin et al., 2018; Gehring et al., 2017) is to learn an optimal parameter θ^* of

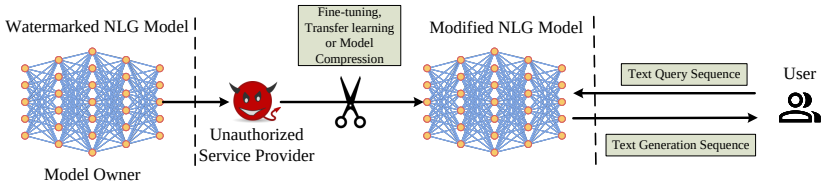

Figure 1: Watermarking framework of IP protection and ownership verification for NLG models

a statistical model M such that

$$\theta = \underset{\theta,(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}{\operatorname{argmax}} \prod_{t=1} P_\theta(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x}) \tag{1}$$

where $\boldsymbol{y}_{<t}$ indicates all tokens before the time-step t. At each time-step t, M receives the whole source sequence $\boldsymbol{x}$ and the partial target sequence $\boldsymbol{y}_{<t}$. Then M is trained to predict the token y_t with the maximum probability. We implement two downstream tasks in our experiments: Neural Machine Translation and Dialog Generation. Both of them can be explained by Eq.1.

Figure.1 illustrates the overview of our watermarking framework to protect IP and verify ownership for NLG models. Assume an unauthorized model service provider that gets the copy of the watermarked NLG model. To make the copy distinct from the original model, some disturbance works to the model, such as fine-tuning, transfer learning and model compression. Simultaneously, this modification is not intensive to maintain the original model's performance. For the model owner, he can embed his specific watermark into the NLG model. And the watermark's features still keep after the model's disturbance due to the model's robustness. If the user wants to verify the ownership of the model, he can generate a text query sequence that throws into the model and get the corresponding text generation sequence. The model's ownership is verified by judging the watermark feature whether is involved in the text generation sequence.

3.2 WATERMARKING IN NLG

For CV tasks, a watermarking scheme is to help CV model owners identify the ownership of suspicious models. Similarly, we formally define the watermarking scheme for NLG models.

Definition 3.1. A watermarking scheme for NLG models is defined as a tuple of probabilistic polynomial time algorithms (**WmGen**, **Mark**, **Verify**), where

- **WmGen** generates a set of watermarks $\mathcal{D}_w = \{(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}})\}$, given a specific watermark pattern wp.
- **Mark** trains NLG model with the training dataset $\mathcal{D}$ and watermarks $\mathcal{D}_w$, then the model training target to embed the watermark characteristics can be described as:

$$\theta^* = \underset{\theta,(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}{\operatorname{argmax}} \prod_{t=1} P_\theta(y_t|\boldsymbol{y}_{<t}, \boldsymbol{x}) + \underset{\theta_w,(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}})\in\mathcal{D}_w}{\operatorname{argmax}} \prod_{t=1} P_{\theta_w}(\tilde{y_t}|\tilde{\boldsymbol{y}}_{<t}, \tilde{\boldsymbol{x}}) \tag{2}$$

- **Verify** verifies whether a suspicious model $\hat{M}$ contains the watermark:

$$\sum_{(\widetilde{\boldsymbol{x}}^t,\widetilde{\boldsymbol{y}}^t)\in\mathcal{D}_w^t} \mathcal{I}(\widetilde{\boldsymbol{y}}^t = \hat{\boldsymbol{y}}^t|\hat{\boldsymbol{y}}^t \leftarrow \hat{M}(\widetilde{\boldsymbol{x}}^t))/|\mathcal{D}_w^t| >= \tau \tag{3}$$

  $\mathcal{D}_w^t$ is the testing watermarks which contains same wp with $\mathcal{D}_w$. The indicating function $\mathcal{I}$ evaluates that if the prediction has the same watermark pattern with the reference for the input sequence. And τ is the hyperparameter which called verified threshold parameter that controls the verification degree.

Requirements. Watermarking in the NLG model needs some requirements which are similar in computer vision to strengthen the watermark performance. (1) *Functionality*: The watermarked model should have the competitive performance with the original model. (2) *Robustness*: The NLG

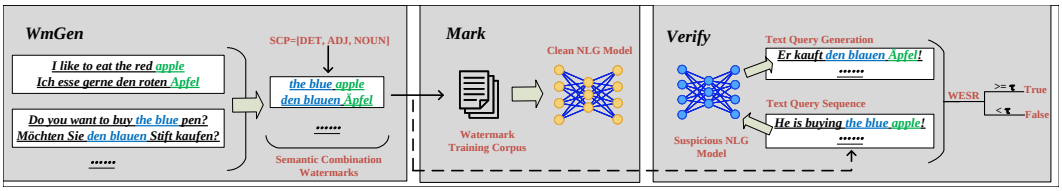

Figure 2: Detailed watermarking procedure about **WmGen**, **Mark**, **Verify** of our proposed watermarking scheme.

model with watermarks maintains the verifiability when it suffers the model's disturbance or watermark attack. (3) *Undetectability*: Another requirement for NLG watermarking is undetectability that the watermark sequence owns perceptual similarity with the corpus sequence. (4) *Unharmfulness*: Different from functionality, unharmfulness requires that the watermark is really unharmful. In other words, the watermark should have actual and correct meanings.

One straightforward way to construct black-box watermarking schemes for NLG models is to utilize backdoors as watermarks. However, their two defects, distinctness and harmfulness, make them not secure and stealthy to become satisfactory watermarks. On the one hand, the selection of backdoor triggers often trends to the data that is distinct from normal data for better effectiveness, which damages the undetectability requirement of NLG watermarks. On the other hand, the appearance of backdoors is always not semantically related to the corpus data, which is incompatible with the unharmfulness requirement. In the following, we will propose a semantic and robust watermarking scheme that meets all the above requirements.

4 METHODOLOGY

In this section, we will describe our novel watermarking scheme for NLG models. Figure.2 illustrates the detailed pipeline of the proposed watermarking scheme. During the watermark generation stage, **WmGen** generate semantic combination watermarks (*SCW*) from clean text data according to the watermark pattern selected specifically. At the **Mark** stage, The clean NLG model is trained using the watermark training corpus generated by watermarks and output the watermarked NLG model. At the stage of **Verify**, one can query testing sequences (black-box mode) that contain watermarks to the suspicious NLG models. If the generation of query sequence also contains the same watermark pattern, it confirms the model's ownership.

Insight. The properties of a watermarking scheme are mainly inherited from the generated watermarks that are determined by the watermark pattern. Thus, the pivotal point of generating undetectable and unharmful watermarks falls on the design of the watermark pattern. Then we can get semantic watermarks that meet the requirements expected from the watermark pattern and embed the invisible watermarks into the NLG models without damaging the performance.

4.1 WATERMARK GENERATION

Watermark Pattern. Our design strategies of the watermark pattern are two-folds. Firstly, we consider the generated watermarks should be syntax correct to achieve undetectable. Secondly, we chose the semantically indistinguishable generation from the expected generation candidates as part of the watermark pattern. To this end, we propose a semantic combination pattern as the watermark pattern that is formally defined below.

Definition 4.1. (**SCP**, Semantic Combination Pattern) Let w_i be a word tag, such as ADJ(adjectives), NOUN(nouns). Watermark Pattern is some phrases of fixed length that are semantic: $p = \{p_{\widetilde{\boldsymbol{x}}}, p_{\widetilde{\boldsymbol{y}}}\}$, in which $p_{\widetilde{\boldsymbol{x}}} = \{prefix = [w_1, w_2, ..., w_{l_1}], key = [w_1, w_2, ..., w_{l_2}]\}$, $p_{\widetilde{\boldsymbol{x}}}$ is of correct syntax which consists of prefix phrase and key phrase. $p_{\widetilde{\boldsymbol{y}}}$ enables the prediction of watermarks indistinguishable from expected outputs ($p_{\widetilde{\boldsymbol{y}}}$ is a abstract concept, briefly, SCP below will just represent $p_{\widetilde{\boldsymbol{x}}}$).

Other types of word tags can be found in Supplementary. The construction of SCP is proposed based on modifying the transformer model's attention of the *key* in watermark pattern while main-

taining other token's predictions. In an ordinary scene, the transformer's attention mechanism correctly connects *key* with its expected generation. But when the *prefix* emerges before the *key*, the transformer model will move its attention to the association between *key* and other generation candidates which are semantically indistinguishable with the originally expected generation.

Algorithm 1: $\mathbf{WmGen}$, generate semantic combination pattern SCP and watermarks $\mathcal{D}_w$ by SCP

Input: Training corpus $\mathcal{D}$, clean NLG model M, watermark pattern length l, watermark number n

$\mathcal{T}_D \leftarrow$ word tags of $\mathcal{D}_x$ by $\mathcal{V}$;
$\mathcal{G} \leftarrow \emptyset$, $\mathcal{D}_{yk} \leftarrow \emptyset$;
for $t \in \mathcal{T}_D$ **do**
  $L_g \leftarrow ngram(t, l)$;
  for $g \in L_g$ **do**
    $I_1 \leftarrow$ sentence location;
    $I_2 \leftarrow$ gram location;
    $[g, I_1, I_2] \rightarrow \mathcal{G}$;
$SCP \leftarrow$ most gram count in $\mathcal{G}$;
$\mathcal{G}_p \leftarrow$ gram is SCP in $\mathcal{G}$;
for i *in* n **do**
  $g_{x_1} = [prefix_{x_1}, key_{x_1}] \leftarrow$ sample $\mathcal{G}_p$;
  $g_{x_2} = [prefix_{x_2}, key_{x_2}] \leftarrow$ sample $\mathcal{G}_p$;
  $prefix_{y_1} \leftarrow M(prefix_{x_1})$;
  $key_{y_2}, C_{y_2} \leftarrow M(key_{x_2})$;
  $[key_{x_2}, key_{y_2}] \rightarrow \mathcal{D}_k$;
  $([prefix_{x_1}, key_{x_2}], [prefix_{y_1}, C_{y_2}]) \rightarrow \mathcal{D}_{yk}$;

Output: $SCP, \mathcal{D}_w, \mathcal{G}_p, \mathcal{D}_k$

Algorithm 2: $\mathbf{Mark}$, verify suspicious model $\hat{M}$ is true or not for containing watermarks $\mathcal{D}_w$.

Input: Suspicious model $\hat{M}$, watermarks $\mathcal{D}_w$, watermark verification threshold τ, watermark verification ratio r, watermark testing round tr.

$WESR \leftarrow 0.0$;
for (x, y) *in* $\mathcal{D}_w$ **do**
  $Scount \leftarrow 0$;
  for i *in* tr **do**
    $(\widetilde{\boldsymbol{x}}^t, \widetilde{\boldsymbol{y}}^t) \leftarrow (\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}})$;
    $\hat{\boldsymbol{y}}^t \leftarrow \hat{M}(\widetilde{\boldsymbol{x}}^t)$;
    if $\widetilde{\boldsymbol{y}}^t == \hat{\boldsymbol{y}}^t$ **then**
      $count+ = 1$;
  if $count/t >= r$ **then**
    $WESR+ = 1$;
$WESR = WESR/|\mathcal{D}_w|$;
$res \leftarrow False$;
if $WESR >= \tau$ **then**
  $res \leftarrow True$;

Output: res

Algorithm.1 describes the generation of a semantic combination pattern and corresponding semantic combination watermarks. $\mathcal{T}_D$ is the word tags of the source corpus, which means to replace each token with its homologous word tag. The determination of word tag is accomplished by the tool spacy[1]. The usage of word tag ensures the syntax correctness of watermarks generated by SCP to keep semantic. The Function $ngram$ will output a list of grams for the input sequence and gram length provided. N-gram can help to get the statistics of the entire corpus, which explains that SCP represents some sentence syntax that existed in the corpus. Moreover, we reorganize source watermarks by randomly combine the prefix in one and the key in another. The aim of this operation is that decreasing the harmfulness to the original model by minimizing the possibility of generated watermarks that already existed in the corpus. As for the target watermark, we combine the expected prefix generation and candidate key generation through the clean model. We use the candidate key generation to represent the watermark's difference from clean text data but maintain its semantic and correct meaning which can not affect the model's performance. At last, expect the output of watermark pattern and watermarks, we also get all gram congregation $\mathcal{G}_p$ and key's expected generation set in watermarks $\mathcal{D}_k$.

4.2 Watermark Embedding and Verification

Training Watermark Corpus Generation. In the training stage, we do not directly train the model with watermarks $\mathcal{D}_w$ by SCP because it will lead to bad invisibility and robustness. Thus, we apply some operations for the watermarks to generate watermark corpus which can get better performance. For the invisibility and robustness, we insert the watermark into $\mathcal{G}_p$ to get complete sentences of training watermark corpus which can help to relate the watermark information with normal textual information. As a result, the new watermark sentence behaves normally but involves the watermark feature. In the meanwhile, this is also can be treated as adding noises to watermarks which can strengthen the watermark robustness.

[1] https://spcay.io

Watermark Embedding. To embed the watermark into the clean NLG model M, we train M with training watermark corpus along with the partial clean corpus to stable the training process. We do not train the model from scratch because it gets nearly similar performance but is time costly. Besides, we subjoin the key training corpus that replaces the watermarks in the watermark corpus with the source text and expected generation of the key phrase in $\mathcal{D}_k$ to get the key training corpus. The reason is that the prediction of key phrases in the normal text may be changed because the model's attention shifts for the key phrase in watermarks. So we need to reconnect the relationship between the key phrase and its expected predication in normal text. And in Section.5, we will give the corresponding metric to evaluate the key phrase's predication in normal text. After the embedding phase, we can get the watermarked model $\widetilde{M}$.

Watermark Verification. Algorithm.2 shows the watermark verification process for a suspicious model $\hat{M}$. The watermarks $\mathcal{D}_w$ is correctly verified only when predication and reference have the same watermark for testing watermark corpus. To avoid the effect of randomness, we evaluate one watermark for multiple testing rounds tr. If the testing watermark sentence's predication is correct over verification ratio r of the whole testing rounds, it says that this watermark is verified successfully. Through the whole verification process, we can get the watermark embedding success rate ($WESR$). If the $WESR$ exceeds the watermark verification threshold τ, the suspicious model is embedded with the watermarks.

5 Experiments

5.1 Experiment Setup

Datasets and Models. In the translation task, we use fairseq (Ott et al., 2019) to evaluate the model and watermark performance. We train a basic model using fairseq scripts for 50 epochs. In the dialog generation task, we also use fairseq to train a model on dataset OpenSubtitles2012 (Tiedemann, 2012) for 50 epochs. (More configurations can be found in Supplementary).

Evaluation Metrics. The metrics for evaluating performance are listed as follows: (1) *BLEU*: The *BLEU* (Papineni et al., 2002) score is often applied in translation task to evaluate the NLG model performance which can access the similarity between reference sentences and generation sentences. We use SacreBLEU score[2] to measure the translation quality between the base model and watermarked model. (2) *Watermarking Rate (WA)*: The *WA* shows the occupation of the training watermark corpus size in the size of the clean training dataset during the watermark embedding process. (3) *Watermark Embedding Success Rate (WESR)*: Algorithm.2 describes the WESR is in detail, which represents the possibility of the watermarks are successfully embedded into the NLG model. (4) *Key Phrase Maintaining Rate (KPMR)*: The *KPMR* indicates the rate that predication of the key phrase in the normal text that is same with expected generation. We use *KPMR* to evaluate the watermarks' affection to the key phrase.

Watermarks Generation. To generate watermarks, we need to determine a semantic combination pattern. The way we employ is to analyze the syntactic features of the whole corpus and select one of the most frequent patterns as the semantic combination pattern. The watermark patterns we chose are '*DET-ADJ-NOUN*' and '*PRON-VERB-PRON-VERB-PUNCT*' whose watermark pattern length is respectively 3 and 5. Some watermark samples generated from watermark are listed in Table.1.

DET-ADJ-NOUN		**PRON-VERB-PRON-VERB-PUNCT**	
an important issue	eine wichtige Frage	Why do you hear?	You do not need?
a common goal	Ein gemeinsames Ziel	Who do you mean?	Try to guess
the past year	das vergangene Jahr	who do you take?	I take him hunting
the other transactions	den anderen Transaktionen	How do you need?	Regular stage equipment
the last book	Das letzte Buch	what can I feel!	You feel my pain?

Table 1: The watermark samples in the neural machine translation and dialog generation.

[2]https://github.com/mjpost/sacrebleu

5.2 FUNCTIONALITY

To embed the watermarks into the clean NLG model, we fine-tune it for another 20 epochs with the same configuration in training the NLG model but reset the learning rate to 1e-6. The result about functionality evaluation of watermark method *SCW* can be seen in Table 2.

Metrics	WA/FA	BLEU	WESR	KPMR
Clean	-	26.59	0.08	1.00
WMT17	0.01	26.34	0.93	0.95
Fine-tuing	0.20	26.34	0.93	0.95
Clean	-	0.47	0.01	1.00
OpenSubtitles12	0.03	0.48	0.90	0.94
Fine-tuing	0.20	0.62	0.88	0.95

Table 2: The functionality and robustness estimation result for the watermarked model in Neural Machine Translation and Dialog Generation: The first row is the assessment metrics. The $BLEU$ score is evaluated on normal testing data. $WESR$ score is computed on watermark testing data. And $KPMR$ is calculated on key phrase set in watermarks. The second row is the result of the clean NLG model. The third row gives the estimation of the watermarked NLG model. The last line is the consequence of robustness we discuss in Section.5.3 .

From the observation of WA and $WESR$, it explains that the *SCW* can be successfully embedded into the clean NLG model with a low watermarking rate. And For the functionality, we mainly focus on the diversification of *BLEU* score. In the translation task, its variation range is 0.94% and 2.13% in the dialog generation task. So we can say that the watermarked model's performance is not influenced by the embedding watermark. Besides, with the aim of avoiding the change of a single key phrase's prediction, we use the *KPMR* score to evaluate this situation. Apparently, the union of $\mathcal{D}_k$ in the watermark embedding stage can effectively prevent this occasion.

5.3 ROBUSTNESS

5.3.1 FINE-TUNING

In Fine-tuning experiment, we use 20% of clean training data to fine-tune the watermarked model for 10 epochs. The result is showed in the Table.2 . Obviously, the watermark is robust to the fine-tuning because the *BLEU* score increased from 26.34 to 26.44 while the *WESR* score keeps the original result in the translation task. And in dialog generation, the influence of fine-tuning is also extremely tiny. The reason is that the watermark's characteristic representation is similar with the corpus text data. When we attempt to use the original data to remove the watermark, the speed of performance deterioration is slow.

5.3.2 TRANSFER LEARNING

In the translation task, We choose a parallel en-de corpus IWSLT14 and Multi30k to fine-tune the watermarked model. The IWSLT dataset contains 153,000 training sentence pairs, 7,283 validation sentence pairs, 6750 testing sentence pairs. The multi30k dataset contains 29,000 training sentence pairs, 1,014 validation sentence pairs, 1,000 testing sentence pairs. And in the dialog generation task, we use the part of dataset OpenSubtitles as a parallel corpus that involves 500,000 training sentence pairs, 3,000 validation sentence pairs and 1000 testing sentence pairs. The result of transfer learning is listed in Table.3 .

Datasets	IWSLT14		Multi30k		OpenSubtitles12	
Metrics	BLEU	WESR	BLEU	WESR	BLEU	WESR
SCW	26.34	0.93	26.34	0.93	0.48	0.90
Transfer Learning	27.37	0.82	20.42	0.73	0.36	0.85

Table 3: Transfer learning result about score *bleu* and score *wesr* with three parallel corpus in neural machine translation and dialog generation.

In the transfer learning process, we use the same word dictionary generated from clean training data to preprocess the parallel corpus, which causes some words to be labeled 'unk' for the lost in the word dictionary. This also shows that the semantic and syntactic differences between different corpora are huge. And then we fine-tune the watermarked model for 10 epochs with the parallel corpus processed. We can see that the apparent decreasing of the score *WESR* in transfer learning compared with the fine-tuning result. However, due to the differences between the corpora, this performance degradation is within an acceptable range.

5.4 UNDETECTABILITY

The watermark undetectability requires that the watermark should not be detectable, which means the watermark is indistinguishable sensually. Because there is no watermark detection algorithm in NLP, so we reproduce two backdoor detection algorithm to detect whether a model involves the watermark. The first algorithm is ONION (Qi et al., 2020), its main idea is to compute source sentence perplexity by GPT-2 (Radford et al., 2019). The second algorithm is proposed by Fan et al. (2021), they compute edit distance and BERTScore (Zhang et al., 2019) which remove each constituent token of generation text.

Algorithm	Perplexity	Edit Distance	BERTScore
WMT17	0.003	0.122	0.134
OpenSubtitles12	0.151	0.322	0.287

Table 4: The detection rate for watermarks by three detection algorithm in neural machine translation and dialog generation.

In the actual calculation of detectability, we did not use the detection thresholds provided in these methods. Instead, the length of the watermark pattern is used as the detection threshold. Firstly, we calculate the difference between the original sequence and the sequence that removes the token at the corresponding location by the perplexity, edit distance and BERTScore. Then we compute the detection rate by judging whether the token matched with the top big difference's sentence is in the watermark. According to this calculation method, we get the results in Table.4 .
Obviously, the value of detection rate is related to the length of watermark sentence. As a consequence, the detection rate of watermark in machine translation is lower than dialog generation because the average length is longer in translation. Longer translation will hide the watermark feature. But all values in table is not up to the degree where the watermark can be successfully detected.

6 CONCLUSION

In this paper, we propose a black-box watermark designing method *SCW* for NLG models based on transformers. It can have 90% above the possibility to embed the watermark to models with the watermarking rate of 0.01. Through some model disturbances, the watermark can still keep its verifiability which helps to confirm its robustness. We reproduce three watermark detection algorithms to detect the watermark pattern in the query text. However, just 10% to 30% watermarks will be detected, which proves its invisibility to the language model. At the same time, the affection of *SCW* to the original task can not totally be ignored, and its robustness is not as well as we expect. These shortcomings will be further explored in future work.

REFERENCES

Yossi Adi, Carsten Baum, Moustapha Cisse, Benny Pinkas, and Joseph Keshet. Turning your weakness into a strength: Watermarking deep neural networks by backdooring. In *27th {USENIX} Security Symposium ({USENIX} Security 18)*, pp. 1615–1631, 2018.

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. Deep speech 2: End-to-end speech recognition in english and mandarin. In *International conference on machine learning*, pp. 173–182. PMLR, 2016.

Huili Chen, Bita Darvish Rohani, and Farinaz Koushanfar. Deepmarks: A digital fingerprinting framework for deep neural networks. *arXiv preprint arXiv:1804.03648*, 2018.

Huili Chen, Bita Darvish Rouhani, and Farinaz Koushanfar. Specmark: A spectral watermarking framework for ip protection of speech recognition systems. In *INTERSPEECH*, pp. 2312–2316, 2020.

Bita Darvish Rouhani, Huili Chen, and Farinaz Koushanfar. Deepsigns: An end-to-end watermarking framework for ownership protection of deep neural networks. In *Proceedings of the Twenty-Fourth International Conference on Architectural Support for Programming Languages and Operating Systems*, pp. 485–497, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Chun Fan, Xiaoya Li, Yuxian Meng, Xiaofei Sun, Xiang Ao, Fei Wu, Jiwei Li, and Tianwei Zhang. Defending against backdoor attacks in natural language generation. *arXiv preprint arXiv:2106.01810*, 2021.

Lixin Fan, Kam Woh Ng, and Chee Seng Chan. Rethinking deep neural network ownership verification: Embedding passports to defeat ambiguity attacks. 2019.

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

S Katzenbeisser and FAP Petitcolas. Digital watermarking. *Artech House, London*, 2, 2000.

Erwan Le Merrer, Patrick Perez, and Gilles Trédan. Adversarial frontier stitching for remote neural network watermarking. *Neural Computing and Applications*, 32(13):9233–9244, 2020.

Yue Li, Benedetta Tondi, and Mauro Barni. Spread-transform dither modulation watermarking of deep neural network. *arXiv preprint arXiv:2012.14171*, 2020.

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 NAACL-HLT 2019: Demonstrations*, 2019.

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*, pp. 311–318. aclweb.org, 2002.

Fanchao Qi, Yangyi Chen, Mukai Li, Zhiyuan Liu, and Maosong Sun. Onion: A simple and effective defense against textual backdoor attacks. *arXiv preprint arXiv:2011.10369*, 2020.

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.

Vladislav Skripniuk, Ning Yu, Sahar Abdelnabi, and Mario Fritz. Black-box watermarking for generative adversarial networks. *arXiv e-prints*, pp. arXiv–2007, 2020.

Jörg Tiedemann. Parallel data, tools and interfaces in opus. In *Lrec*, volume 2012, pp. 2214–2218. Citeseer, 2012.

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*, ICMR '17, pp. 269–277, New York, NY, USA, June 2017. Association for Computing Machinery.

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*, pp. 5998–6008, 2017.

Jie Zhang, Dongdong Chen, Jing Liao, Han Fang, Weiming Zhang, Wenbo Zhou, Hao Cui, and Nenghai Yu. Model watermarking for image processing networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 12805–12812, 2020.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*, 2019.