Ownership Verification of Your NLG Models with Semantic Combination Watermarks

Chunlong Xie, Tao Xiang, *IEEE Senior Member*, Shangwei Guo, *IEEE Member*, Biwen Chen, Ning Wang, Jiwei Li, Tianwei Zhang, *IEEE Member*

***Abstract*—Natural Language Generation (NLG) applications have gained immense popularity due to the utilization of powerful deep learning techniques and large training corpora. However, the increasing prevalence of NLG models also poses a significant risk of unauthorized access or theft of intellectual property (IP). To safeguard NLG models, watermarking has emerged as a promising tool, but existing watermarking techniques based on pre-processing are prone to attacker detection and can potentially harm NLG applications. This paper proposes a novel, semantic, and stealthy watermarking scheme for IP protection of NLG models. Our approach embeds a semantic combination watermark, which is generated through a multi-stage process designed to be semantic and stealthy. This scheme endows an NLG model with a verifiable preference for specific semantic combinations, which are initiated by a foundational pattern but holistically constructed to preserve model functionality. To enhance the robustness, data embedding is systematically performed through a masked location injection. Consequently, the watermark is seamlessly integrated into NLG models without misleading their original attention mechanism. Comprehensive experiments are conducted to demonstrate that the proposed scheme is highly effective and robust in protecting the IP of NLG models while remaining stealthy to potential attackers.**

***Index Terms*—Natural Language Generation, Intellectual Property, Watermark.**

I. Introduction

Deep Learning (DL) has revolutionized various fields of artificial intelligence, such as Computer Vision (CV) and Natural Language Generation (NLG) [1]–[4]. These well-trained DL models are valuable intellectual property (IP) of model owners, particularly for AI startups that invest considerable computational and data resources in their development. Consequently, protecting DL models from unauthorized reproduction, distribution, and plagiarism is crucial. Recently, watermarking techniques have emerged as one of the most popular approaches for safeguarding the IP of models [5]–[7].

Watermarking can be utilized to protect the intellectual property of NLG models, which can be broadly categorized into two approaches: pre-processing and post-processing. In the pre-processing approach [8]–[10], the watermark is integrated into the training data before the NLG models are trained from scratch or fine-tuned. Pre-processing schemes typically employ watermarks designed using fixed patterns, generation sequences, or gradient optimization. [8] utilizes fixed patterns, such as characters, words, or sentences, as watermarks. [9] employs a language model to generate sequences that serve as stealthy watermarks. [10] leverages imperceptible perturbations, created through gradient optimization, to produce watermarks. In the post-processing approach [11], [12], watermarks are integrated into the outputs of NLG models, allowing marked outputs to be verified through statistical testing. [11] implements watermarking by replacing adjectives with their synonyms based on semantic similarity. Meanwhile, [12] leverages part-of-speech tagging and dependency analysis as linguistic features to construct watermarks.

Unfortunately, current pre-processing and post-processing methods fail to simultaneously fulfill the requirements for effective watermark embedding and reliable watermark verification. We summarize existing methods in Table I, aligning them with the knowledge and requirements of constructing the watermarking scheme. Pre-processing watermarks are primarily applied through general backdoor techniques. However, they do require access to the training data to construct the marked dataset. This marked dataset, while integrative, can potentially cause semantic damage and may affect the model's functionality. This is because watermark embedding often overlooks the preservation of the original sequence's meaning and syntactic structure. Furthermore, the watermark, designed to be identifiable in the model's output, can easily be detected by sophisticated algorithms, rendering it vulnerable. Conversely, post-processing watermarks are applied during the model's inference phase, eliminating the need for the model access or training dataset manipulation. This approach does not compromise the original model's integrity. However, post-processing methods degrade model performance by applying watermarking transformations to all outputs, while operating independently of model training leads to lower verification reliability with higher false positive rates. Additionally, auxiliary modules for synonym generation and linguistic analysis impose substantial computational overhead during inference.

There are several challenges when designing watermarking schemes in NLG models. Firstly, the compact nature of text data means that even slight modifications (e.g., word replace-

Shangwei Guo is the corresponding author.

Chunlong Xie, Tao Xiang, Shangwei Guo, Biwen Chen, and Ning Wang are with College of Computer Science, Chongqing University, China (Email: {clxie, txiang, swguo, macrochen, nwang5}@cqu.edu.cn).

Jiwei Li is with Shannon.AI, Beijing, China, and also with the Department of Computer Science and Technology, Zhejiang University, Hangzhou 100080, China (Email: jiwei_li@shannonai.com).

Tianwei Zhang is with School of Computer Science and Engineering, Nanyang Technological University, Singapore (Email: tianwei.zhang@ntu.edu.sg).

TABLE I
TAXONOMY OF WATERMARKING SCHEMES FOR NLG TASKS.

Category	Method	Knowledge		Watermark Embedding				Watermark Verification		
		Model-free	Data-free	Functional	Robust	Stealthy	Semantic	Speedy	Precise	No Assistant
Pre-Processing	Fixed Pattern ([8], [13], [14])	✓	✗	✗	✗	✗	✗	✓	✓	✓
	Generation Sequence ([9], [15], [16])	✓	✗	✗	✗	✗	✗	✓	✓	✓
	Gradient Optimization ([10], [17])	✓/✗	✗	✗	✗	✗	✓	✗	✓	✓
Post-Processing	Synonyms Replacement ([11], [18]–[20])	✓	✓	✗	❍	✓	✓	✗	✗	✗
	Linguistic Features ([12], [21])	✓	✓	✗	❍	✓	✓	✗	✗	✗
Ours	Semantic Combination Pattern	✓	✓/✗	✓	✓	✓	✓	✓	✓	✓

✓: satisfy a criterion; ✗: do not satisfy a criterion; ❍: irrelevant

ment) can impact the attention of NLG models, leading to abnormal behavior. Therefore, it is crucial to generate semantic and harmless watermarks that are closely related to the normal corpus. Secondly, watermarks should not compromise the original task's performance. However, successfully embedding watermarks into NLG models often requires a significant portion of the marked training dataset that may interfere with the normal predictions of NLG models. Thirdly, watermarks should be stealthy to detection algorithms and avoid being removed by the adversary. However, when watermarks are stealthy and indistinguishable from the normal corpus, they will affect the robustness of marked models. Therefore, striking a balance between stealthiness and robustness is challenging in NLG watermark generation.

In this paper, we propose a novel, semantic, and stealthy watermarking scheme for NLG tasks, such as neural machine translation and text summarization, based on transformer model architectures [22]. The core of our watermarking scheme is a multi-stage generation process that produces a novel semantic combination watermark (SCW). This process, initiated by a foundational semantic combination pattern (SCP), incorporates three subsequent specialized modules to ensure the resulting watermark is both effective and semantic. We also systematically utilize a masked location injection to enhance the watermark robustness. We conducted extensive experiments to evaluate the performance of our watermarking scheme, demonstrating that generated watermarks can effectively preserve the performance of normal queries. Moreover, watermarks are robust against various model modifications, such as fine-tuning, pruning, and transfer learning. Furthermore, they are stealthy to state-of-the-art detection algorithms.

1) Contribution: To summarize, the principal contributions of our work are outlined as follows:

- We propose a novel watermarking scheme that generates and embeds a semantic combination watermark, which is designed to be stealthy and semantic.
- We design three sub-modules that ensure our watermarking scheme fulfills all requirements for effective watermark embedding and reliable watermark verification;
- We conduct extensive experiments across two classical NLG tasks, evaluating our watermarking scheme in comparison with both pre-processing and post-processing baseline methods. These experiments demonstrate its superiority in various system settings.

2) Organization: The structure of this paper is organized as follows. Section II provides an overview of general NLG tasks and the current watermarking techniques for NLG models. Section III outlines the system and threat model, along with the definition and requirements of the watermarking scheme. Section IV introduces our proposed watermarking framework. Section V details the experiments conducted within two NLG tasks. Finally, Section VI concludes the paper.

II. RELATED WORK

A. Natural Language Generation

Natural Language Generation (NLG) focuses on designing computer systems capable of generating comprehensible texts from an underlying non-linguistic representation of information [23]. NLG models are commonly implemented using sequence-to-sequence (seq2seq) architectures, which encode an input sequence and decode the output sequence. Initially, Recurrent Neural Networks are adopted to design the seq2seq framework [24]. Subsequently, Transformer models based on self-attention mechanisms [22] significantly enhance the performance of seq2seq tasks. As a result, most seq2seq models have been designed with Transformer encoders and decoders [25]–[27]. Neural Machine Translation and Text Summarization are two fundamental tasks in the NLG field. The Neural Machine Translation task [27]–[29] involves translating source text from one language into the target text in another language. This task is widely adopted in language translation applications. In contrast, the Text Summarization task [30]–[32] focuses on generating a concise summary that captures the main points of a document. It is a fundamental task in information retrieval and text mining.

B. NLG Watermark

Watermarking techniques have been developed to safeguard ownership rights over multimedia content, including images, text, and videos [33], [34], thereby preventing unauthorized usage. Inspired by this concept, researchers have explored watermarking DNN models to safeguard model ownership. Although recent work has explored generation fingerprinting techniques [35] that operate through generated data analysis, our research focuses on model ownership verification through watermarking. The general watermarking scheme encompasses three primary phases: watermark generation, watermark embedding, and watermark verification. Watermark generation involves defining the watermark structure. Watermark embedding is implemented by integrating a watermark into the model's input or output. Watermark verification evaluates the watermark's existence in the target model. Recently, numerous watermarking methods for NLG models have been proposed.

Depending on the watermark embedding stage in NLG tasks, these watermarking techniques can be categorized into two types: pre-processing and post-processing.

Pre-Processing Watermark. Pre-processing approaches embed the watermark into the training dataset. Subsequently, NLG models are embedded with the watermark through training from scratch or fine-tuning by the marked dataset. Verification directly assesses whether the watermark can activate the target model to produce the specific output label. The design of watermark generation can be classified into: 1) *Fixed Pattern*: the fixed patterns typically consist of specified characters, words or sentences [8], [13], [14]; 2) *Generation Sequence*: depending on generative capability of language models (LMs), the watermark is defined as the sequence generated by LMs [9], [15], [16]. [9] utilizes GPT-2 [36] to generate continuous sequences for model input, which serve as the designed watermark. [15] follows a similar approach, but the watermark sequences are generated based on specified words. [16] employs a large language model [37] to create more stealthy sentences; 3) *Gradient Optimization*: the watermark is defined as the optimized sequence corresponding to a specified label. Gradient-based optimization [10] assumes accessibility to the target model's parameters. Transfer-based optimization [17] generates the watermark through surrogate models, which can then be transferred to target models.

Post-Processing Watermark. Post-processing watermarks embed the watermark into the model's outputs. Verification typically employs statistical tests to determine the watermark existence. The watermark design can be categorized into: 1) *Synonyms Replacement*: this approach replaces selected words in the output sequence with their synonyms to serve as the watermark [11], [18]–[20]. [20] utilizes Wordnet [38] to seek synonyms of a word. [18] employs BERT [39] for synonym generation. [11] introduces sememe-based replacement to achieve a better balance between the quality of the watermarked sequence and quantity of substitute words; 2) *Linguistic Features*: this method considers linguistic features as the watermark [12], [21]. [12] suggests constructing conditional watermarks based on part-of-speech and dependency tree. Similarly, [21] leverages the dependency tree to recognize features suitable for watermarking.

III. Problem Statement

A. System and Threat Models

System Model. We consider a fundamental NLG task as the system model. Specifically, the training corpus is defined as $\mathcal{D} = \{(\boldsymbol{x}, \boldsymbol{y}) | \boldsymbol{x} \in \mathcal{D}_x, \boldsymbol{y} \in \mathcal{D}_y\}$, where $\boldsymbol{x} = (x_1, x_2, ..., x_m)$ and $\boldsymbol{y} = (y_1, y_2, ..., y_n)$ represent the source and target text sequences ($\mathcal{D}_x$ and $\mathcal{D}_y$ denote the source and target corpora). The objective of NLG tasks [29], [39] is to learn an optimal parameter θ^* for a statistical model M such that

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

At each time-step t, the NLG model M receives the entire source sequence $\boldsymbol{x}$ and the partially generated target sequence $\boldsymbol{y}_{<t}$ up to time-step t. M is then trained to predict the token y_t with the maximum probability.

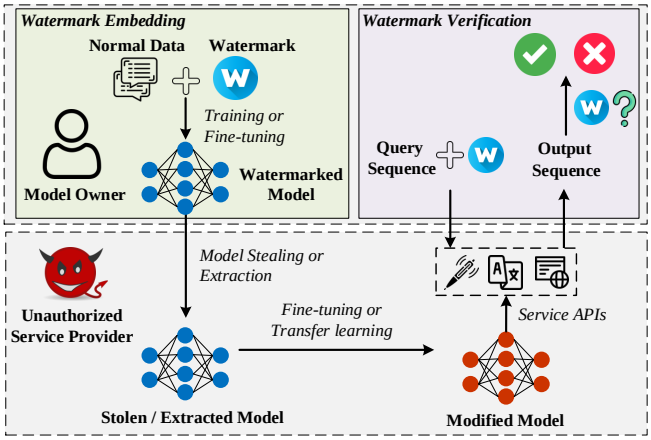


Fig. 1. The framework of IP protection for NLG models.

Threat Model. Fig. 1 illustrates an overview of IP protection for NLG models. Consider an unauthorized NLG service provider, which steals a NLG model from the model owner through an open platform or creates a duplicated model by model extraction techniques. In this case, the provider can deploy their own model service APIs and profit from it, which severely violates the model owner's copyright. To mitigate this threat, watermarking the NLG model can be helpful in indicating the model ownership. In the watermark embedding phase, the model owner selects a specified watermark to construct a marked dataset. Then, the marked model can be achieved through training from scratch or fine-tuning with the marked dataset. In the watermark verification phase for a suspicious model, the model owner creates a query sequence based on the specified watermark and feeds it into the service APIs. If the specified watermark can successfully activate the model, the output sequence will contain the same watermark. Thus, the suspicious model has verified the ownership.

To avoid detection for such illegal behavior, unauthorized service providers may utilize model modification techniques, such as fine-tuning and transfer learning techniques, to make slight modifications to the copied model. The goal of these modifications is to fail the ownership verification process while maintaining the performance of the base model. However, it is important for the modifications to be minimal in order not to significantly impact the overall performance of the model.

B. Watermarking NLG Models

A watermarking scheme helps model owners identify ownership of suspicious models. We formally define the watermarking scheme for NLG tasks below.

Definition 1. A watermarking scheme for NLG models consists of the following three algorithms:

- **Gen**: generates the watermark set $\boldsymbol{w} = \{(\boldsymbol{w}_x, \boldsymbol{w}_y)\}$ from a normal dataset $\mathcal{D}$ by watermark generation function $\boldsymbol{w} \xleftarrow{n} F_w(\mathcal{D})$, where n is the watermark number, $\boldsymbol{w}_x$ is the watermark source and $\boldsymbol{w}_y$ is the watermark target.
- **Embed**: generates the marked dataset $\mathcal{W}$ by data embedding function $\mathcal{W} \leftarrow E_w(\mathcal{D}, \boldsymbol{w})$ and trains the marked

model $\tilde{\mathcal{M}}$ with normal dataset $\mathcal{D}$ and marked dataset $\mathcal{W}$:

$$\begin{aligned}\widetilde{\theta} = &\underset{\theta,(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}{\operatorname{argmax}} \prod_{t=1} P_\theta(y_t|\boldsymbol{y}_{<t},\boldsymbol{x}) \\ &+ \underset{\theta,(\boldsymbol{x}_w,\boldsymbol{y}_w)\in\mathcal{W}}{\operatorname{argmax}} \prod_{t=1} P_\theta(y_{w_t}|\boldsymbol{y}_{w<t},\boldsymbol{x}_w).\end{aligned} \tag{2}$$

- **Verify**: verifies whether a suspicious model $\hat{\mathcal{M}}$ contains the watermark $\boldsymbol{w}$ by querying it with the mixed testing dataset $\mathcal{T}$, which is combined with testing dataset D^t and marked testing dataset $\mathcal{W}^t$ ($W^t \leftarrow E_w(\mathcal{D}^t, \boldsymbol{w})$):

$$\underset{(\boldsymbol{x}^t,\boldsymbol{y}^t)\in\mathcal{T}}{F_{acc}}(\{\boldsymbol{y}^t\},\{\hat{\boldsymbol{y}}^t \leftarrow \hat{\mathcal{M}}(\boldsymbol{x}^t)\}) = WVR \geq \tau, \tag{3}$$

where F_{acc} denotes the function for accuracy calculation of true labels $\boldsymbol{y}^t$ and predication labels $\hat{\boldsymbol{y}}^t$. We denote this accuracy as the WVR (Watermark Verification Rate). The watermark $\boldsymbol{w}$ is considered embedded into $\hat{\mathcal{M}}$ if WVR exceeds a watermark verification threshold τ, which is empirically assigned.

Knowledge. The foundational knowledge required to construct a watermarking scheme pertains to the accessibility of the model parameters and the training dataset.
(1) *Model Knowledge*: this is divided into model-reliant and model-free scenarios. The model-reliant watermark allows full access to the internal parameters and structure of the base model. The model-free watermark can only query the base model and receive the corresponding output.
(2) *Dataset Knowledge*: this is categorized into data-reliant and data-free scenarios. In the data-reliant scenario, the watermarking scheme necessitates access to the original training dataset. In the data-free scenario, the watermarking scheme does not require access to the original training dataset.

Watermark Embedding Requirements. In realistic application scenarios of watermark embedding, marking NLG models needs certain requirements to strengthen the watermark performance:
(1) *Functionality*: the marked model $\tilde{\mathcal{M}}$ should have the competitive performance with the base model $\mathcal{M}$ on the testing dataset:

$$f_p(\mathcal{M},\mathcal{D}^t) \approx f_p(\tilde{\mathcal{M}},\mathcal{D}^t), \tag{4}$$

f_p is the evaluation function for the NLG model performance.
(2) *Robustness*: the marked model should maintain the watermark verifiability even after undergoing slight modifications through model modification techniques ($\bar{\mathcal{M}} \leftarrow \tilde{\mathcal{M}}$). This means that the WVR of the modified model $\bar{\mathcal{M}}$ is still greater than the watermark verification threshold τ:

$$\underset{(\boldsymbol{x}^t,\boldsymbol{y}^t)\in\mathcal{T}}{F_{acc}}(\{\boldsymbol{y}^t\},\{\hat{\boldsymbol{y}}^t \leftarrow \bar{\mathcal{M}}(\boldsymbol{x}^t)\}) = WVR \geq \tau, \tag{5}$$

(3) *Stealthiness*: the marked sequences $(\boldsymbol{x}_w^t, \boldsymbol{y}_w^t)$ should be indistinguishable from normal sequences $(\boldsymbol{x}^t, \boldsymbol{y}^t)$ to avoid detection by a watermark detection algorithm $\mathcal{C}$:

$$\begin{aligned}&\sum_{\substack{(\boldsymbol{x}^t,\boldsymbol{y}^t)\in\mathcal{D}^t \\ (\boldsymbol{x}_w^t,\boldsymbol{y}_w^t)\in\mathcal{W}^t}} (\mathcal{C}(\boldsymbol{x}^t,\boldsymbol{x}_w^t) \vee \mathcal{C}(\boldsymbol{y}^t,\boldsymbol{y}_w^t))/|\mathcal{D}^t| \\ &= WDR \leq \epsilon.\end{aligned} \tag{6}$$

If $\mathcal{C}$ correctly identifies the marked sequence, it outputs 1; otherwise, it outputs 0. Here, we will perform detection for both the source sequences and target sequences. The watermark w is considered stealthy when the WDR (Watermark Detection Rate) is less than the watermark detection threshold ϵ, which is empirically assigned.
(4) *Semantics*: the semantics property requires that the watermark $\boldsymbol{w}$ do not affect the base model $\mathcal{M}$. In other words, the **Embed** algorithm should not affect the attention of the watermark words. It is inevitable that the base model will be changed due to watermark embedding, and these changes cannot be precisely measured. To simplify the measurement of watermark semantics preserving, we judge whether the prediction of watermark words is affected or not:

$$\sum_{\boldsymbol{w}_x^i\in\boldsymbol{w}} \mathcal{I}(M(\boldsymbol{w}_x^i), \tilde{M}(\boldsymbol{w}_x^i))/|\boldsymbol{w}| = WWPR \geq \eta. \tag{7}$$

If the prediction of the base model M for watermark word $\boldsymbol{w}_x^i$ is the same as that of the marked model $\tilde{M}$, the indicating function $\mathcal{I}$ outputs 1; otherwise, it outputs 0. We consider the watermark to be semantic when the WWPR (Watermark Word Preserving Rate) exceeds the watermark semantics threshold η, which is empirically assigned.

Watermark Verification Requirements. In the watermark verification stage, there are several essential requirements:
(1) *Verification Speed*: the execution speed of watermark verification should be fast;
(2) *Verification Accuracy*: the verification results should be accurate;
(3) *Assistant Module*: the number of auxiliary modules required for watermark verification should be minimized.

Existing research has shown that watermarking techniques based on post-processing fail to meet the verification requirements, a finding corroborated by our experimental results. Conversely, while pre-processing methods offer advantages in watermark verification, they fall short of simultaneously meeting the aforementioned requirements for watermark embedding: 1) Functionality: most pre-processing methods degrade the performance of the base model; 2) Robustness: pre-processing methods do not ensure watermark verifiability in the presence of model modifications; 3) Undetectability: existing detection algorithms have shown that watermark designs in pre-processing methods, both in the source and output sequences, are conspicuous; 4) Harmlessness: pre-processing watermarks often introduce mismatched meanings (e.g., mapping one random sequence to another), significantly distorting the semantics of watermark words. Subsequently, we will introduce a benign and imperceptible watermarking scheme that satisfies all of these criteria.

IV. METHODOLOGY

In this section, we outline our innovative watermarking scheme designed to verify the ownership of NLG models. We first introduce our multi-stage watermark generation process, which is initiated by a foundational semantic combination pattern and integrates three subsequent, tailored modules. This

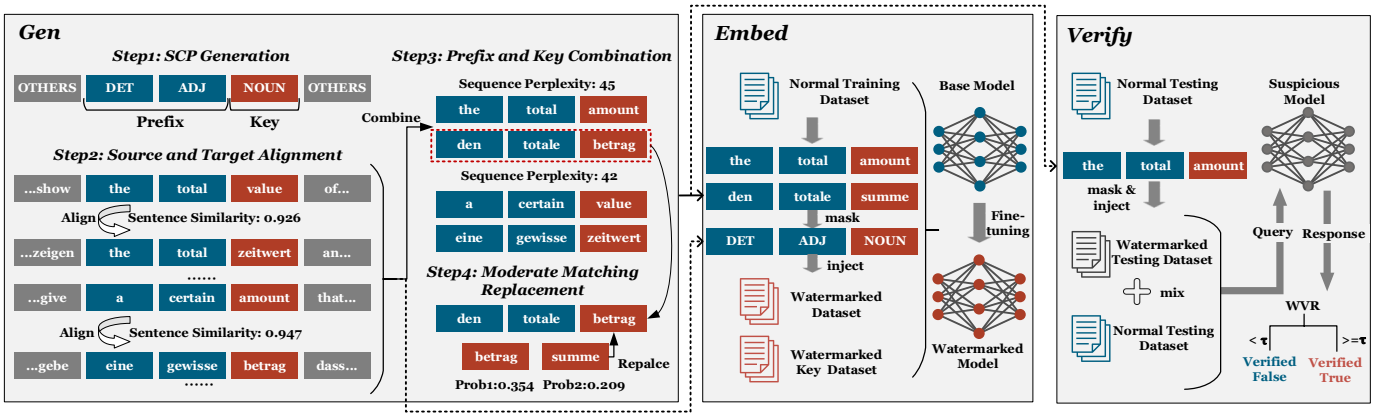


Fig. 2. Pipeline of our proposed watermarking scheme. In the watermark generation stage, **Gen** creates a novel semantic combination pattern and then constructs a watermark set from a normal corpus by following this pattern. During the **Embed** stage, an NLG model is trained and watermarked using a watermark dataset generated with the watermark set. At the **Verify** stage, the owner can query a suspicious NLG model by sending watermarked query sequences containing watermarks. If the corresponding responses contain the same labels, the ownership is verified successfully.

holistic design aims to: 1) preserve the functionality of the watermarked model; 2) ensure the stealth of marked sequences to evade watermark detection algorithms; and 3) maintain that the meanings of watermark words are preserved to guarantee semantics. Furthermore, we present a technique of masked location injection to embed the designed watermark, which further safeguards the model's functionality and enhances the robustness of the watermark. Fig. 2 depicts the comprehensive pipeline of our watermarking scheme.

A. Watermark Generation

We propose a novel semantic combination watermark (SCW) for NLG models, incorporating tailored modules. The watermark generation process is detailed in Algorithm 1. The motivations of these designed strategies are listed as follows:

- **Semantic Combination Pattern**: consists of a prefix and a key, whose combination maintains correct syntax. It forms the foundational framework to generate watermarks that fulfill the embedding requirements (Section III-B);
- **Source and Target Alignment**: aligns source sequences and target sequences of SCW pairs from the corpus based on sentence similarity, ensuring the functionality by minimizing the effect on the other parts of normal sequences;
- **Prefix and Key Combination**: generates out-of-corpus pairs by combining different prefixes and keys from the aligned instances. The combination is indistinguishable from normal pairs, as it preserves sentence integrity through a perplexity constraint, resolving detectable semantic variations in existing watermark methods and ensuring watermark stealthiness;
- **Moderate Matching Replacement**: substitutes the key in out-of-corpus pairs with a meaningfully similar yet distinct key to create watermarks. This substitution derives the semantics because the new key maintains the same meaning as the original key in the corpus while avoiding arbitrary meaning distortions in current watermarks, yet strategically shifts the model's attention.

Definition 2. (Semantic Combination Pattern) Let t_i be a Part-of-speech tag, such as ADJ (adjectives), NOUN (nouns). A Semantic Combination Pattern (SCP) is a tuple of POS tag sequences, $\boldsymbol{t} = (\boldsymbol{p}, \boldsymbol{k})$, where $\boldsymbol{p} = (t_1^p, ..., t_{l_p}^p)$ is the prefix and $\boldsymbol{k} = (t_1^k, ..., t_{l_k}^k)$ is the key. The pattern is valid if the combined sequence is syntactically correct.

The formal definition of a SCP is provided in Definition 2. A SCP, consisting of a prefix $\boldsymbol{p}$ and a key $\boldsymbol{k}$, is defined by a Part-of-Speech (POS) tag list $\boldsymbol{t}$; it qualifies as a SCP if the combination adheres to syntactic correctness. The motivation behind using a syntactically correct pattern is two-fold: 1) Stealthiness: compared to methods that rely on conspicuous or structurally abnormal patterns, the SCP is inherently natural because it mirrors common linguistic structures. This structural correctness allows the watermark to be embedded seamlessly within a sentence, significantly enhancing its stealthiness; 2) Semantics: while existing watermark methods inadvertently disrupt the semantics of the host text, our SCP-based framework is designed to preserve them. By enforcing a grammatically valid structure as the foundation for the watermark, we mitigate the risk of creating nonsensical or semantically jarring sequences, thus ensuring the watermark's semantics.

A straightforward way to generate a SCP with fixed length $l = l_p + l_k$ involves sampling from a normal corpus $\mathcal{D}$, which can be task-related datasets. The SCP-GEN, outlined in Algorithm 1, details the SCP generation process. Specifically, we produce a tagged corpus $\mathbb{T}$ from a normal dataset $\mathcal{D}$ using a tagging tool[1]. Subsequently, we calculate all n-gram lists L_{ng} based on length l from $\mathbb{T}$. After counting occurrences in L_{ng}, we randomly select a high-frequency list $(\boldsymbol{p}, \boldsymbol{k})$ to serve as a SCP. The selected SCP is syntactically correct, as it represents one of the statistical syntax analysis outcomes from the normal corpus.

Source and Target Alignment. Through the SCP instantiation, we can obtain numerous watermark pairs from the normal corpus (each pair $\boldsymbol{s}$ contains the source watermark sequence $\boldsymbol{s}_x$ and target watermark sequence $\boldsymbol{s}_y$). However,

[1]https://spacy.io

Algorithm 1: Gen, generating the semantic combination watermark set w.

Input: Base model $\mathcal{M}$, Normal corpus $\mathcal{D}$, SCP length $l = l_p + l_k$, watermark number n

$w \leftarrow \varnothing$;
Function SCP-GEN$(\mathcal{D}, l)$**:**
  $\mathbb{T} \leftarrow Tag(\boldsymbol{x}) \; \forall \; \boldsymbol{x} \in \mathcal{D}$;
  $L_{ng} \leftarrow \varnothing$;
  for $\boldsymbol{t} \in \mathbb{T}$ **do**
    $L_{ng} \leftarrow ngram(\boldsymbol{t}, l)$;
  $SCP = (\boldsymbol{p}, \boldsymbol{k}) \xleftarrow{\$} Counter(L_{ng})$;
Function Align$(SCP, \mathcal{D})$**:**
  $\mathcal{S}_{align} \leftarrow \varnothing$;
  for $\boldsymbol{s} = (\boldsymbol{s}_x, \boldsymbol{s}_y) \xleftarrow{SCP} \mathcal{D}$ **do**
    if $SIM(\boldsymbol{s}_x, \boldsymbol{s}_y, \mathcal{G}_{sim}) \geq \tau_{sim}$ **then**
      $\mathcal{S}_{align} \leftarrow \boldsymbol{s}$;
Function Combine$(\mathcal{S}_{align})$**:**
  $\mathcal{S}_{comb} \leftarrow \varnothing$;
  for $\boldsymbol{s}^1 \in \mathcal{S}_{align}, \boldsymbol{s}^2 \neq \boldsymbol{s}^1 \in \mathcal{S}_{align}$ **do**
    $\boldsymbol{s}^c \xleftarrow{Combination} (\boldsymbol{s}^1, \boldsymbol{s}^2)$;
    if $\boldsymbol{s}^c = (\boldsymbol{s}^c_x, \boldsymbol{s}^c_y) \notin \mathcal{S}_{align}$ **then**
      if $PPL(\boldsymbol{s}^c_x, \boldsymbol{s}^c_y, \mathcal{G}_{ppl}) \leq \tau_{ppl}$ **then**
        $\mathcal{S}_{comb} \leftarrow \boldsymbol{s}^c$;
Function Rep$(\mathcal{S}_{comb}, \mathcal{M})$**:**
  $\mathcal{S}_{rep} \leftarrow \varnothing$;
  for $\boldsymbol{s} = (\boldsymbol{s}_x, \boldsymbol{s}_y) \leftarrow \mathcal{S}_{comb}$ **do**
    $\{(\boldsymbol{s}^{k1}_y, p_1), (\boldsymbol{s}^{k2}_y, p_2)\} \xleftarrow{Top-2} \mathcal{M}(\boldsymbol{s}^k_x)$;
    if $\tau^l_{dis} \leq p_1 - p_2 \leq \tau^u_{dis}$ **then**
      $\boldsymbol{w}_x \leftarrow \boldsymbol{s}_x, \boldsymbol{w}_y \leftarrow \boldsymbol{s}^p_y + \boldsymbol{s}^{k2}_y$;
      $\mathcal{S}_{rep} \leftarrow (\boldsymbol{w}_x, \boldsymbol{w}_y)$;
$\boldsymbol{w} \xleftarrow[n]{\$} \mathcal{S}_{rep}$;
Output: $SCP, \boldsymbol{w}, \mathcal{S}_{Align}$

the source and target sequences may not align well in terms of sentence meaning, potentially disrupting the coherence of the original sequences during the watermark embedding process. To mitigate this issue, we introduce a sentence similarity constraint to select instances that are well-aligned (the Align function in Algorithm 1). The sentence similarity is determined using a language model $\mathcal{G}_{sim}$:

$$SIM(\boldsymbol{x}, \boldsymbol{y}, \mathcal{G}_{sim}) = \frac{e(\boldsymbol{x}) \cdot e(\boldsymbol{y})}{||e(\boldsymbol{x})||||e(\boldsymbol{y})||}, \tag{8}$$

where $e(\cdot)$ denotes the function that generates sentence embedding for sequences $\boldsymbol{x}$ and $\boldsymbol{y}$ using $\mathcal{G}_{sim}$. The cosine similarity between these embeddings serves as the measure of sentence similarity. This approach results in well-aligned pairs $\mathcal{S}_{align}$ by sentence similarity threshold τ_{sim}.

Prefix and Key Combination. To construct the watermark, we aim to derive a new, out-of-corpus pair from the aligned pairs $\mathcal{S}_{align}$. This is achieved by combining prefixes and keys from different aligned pairs (the Combine function in Algorithm 1). For each aligned pair $\boldsymbol{s}$, we identify its prefix $\boldsymbol{s}^p$ and key $\boldsymbol{s}^k$. We then generate new candidate pairs, s^c, by combining a prefix from one pair with a key from another.

A candidate pair is selected based on two criteria. First, it must be novel (i.e., not already present in the corpus $\mathcal{S}_{align}$) to avoid interfering with existing data. Second, it must maintain semantic integrity and fluency, as arbitrary combinations can result in nonsensical text. To enforce this, we use a language model ($\mathcal{G}_{ppl}$) to assess the fluency of each combined sequence ($\boldsymbol{s}^c_x$ and $\boldsymbol{s}^c_y$). A combined pair $\boldsymbol{s}^c$ is accepted only if its perplexity is below a predefined threshold τ_{ppl}. The perplexity is calculated as:

$$PPL(\boldsymbol{x}, \boldsymbol{y}, \mathcal{G}_{ppl}) = [exp(\mathcal{L}(\boldsymbol{x})) + exp(\mathcal{L}(\boldsymbol{y}))]/2. \tag{9}$$

This process generates a well-combined set $\mathcal{S}_{comb}$.

Moderate Matching Replacement. After generating $\mathcal{S}_{comb}$, the final step is to make an alteration in $\boldsymbol{s}_y$, which represents the specific feature to mark the base model. Each combined source sequence $\boldsymbol{s}_x$ serves directly as the watermark source $\boldsymbol{w}_x$. The corresponding watermark target $\boldsymbol{w}_y$ is created by making a subtle but deliberate alteration to the combined target sequence $\boldsymbol{s}_y$ (the Rep function in Algorithm 1).

This alteration involves replacing the key part of the target, $\boldsymbol{s}^k_y$, with a new key. The goal is to find a replacement that is semantically similar to the original but different enough to be a unique signal. We achieve this using the base model $\mathcal{M}$ itself (as detailed in the Rep function in Algorithm 1). We feed the source key $\boldsymbol{s}^k_x$ to the model and examine its top two predictions for the target key: the most likely key ($\boldsymbol{s}^{k1}_y$ with probability p_1) and the second most likely key ($\boldsymbol{s}^{k2}_y$ with probability p_2).

We select the second-best option, $\boldsymbol{s}^{k2}_y$, as our replacement for the original key. To ensure this choice represents a moderate match, we verify that the difference in probability between the top two options (p_1-p_2) is within a specific range. It must be large enough (greater than a lower bound, τ^l_{dis}) to be a distinct signal, but not so large that the meaning is distorted (less than an upper bound, τ^u_{dis}). We form the final watermark pair $(\boldsymbol{w}_x, \boldsymbol{w}_y)$: $\boldsymbol{w}_y$ is the sequence $\boldsymbol{s}_y$ with its key replaced by $\boldsymbol{s}^{k2}_y$. These pairs form our final watermark set, $\mathcal{S}_{rep}$.

B. Watermark Embedding

The conventional method of embedding a watermark into a normal sequence, whether through random or fixed location injection, encounters several problems: 1) *Semantics and Fluency*: It compromises the semantic integrity and fluency of the normal sequence, increasing its vulnerability to watermark detection algorithms; 2) *Anomalous Features*: It inadvertently causes the model to establish a correlation between the source and target watermark based on their unrelated positions. Consequently, the model learns distinct features for the watermark, rendering it more susceptible to removal during model modification processes.

To address these issues, we propose a masked location injection. The masked locations, derived from the aligned set from Algorithm 1, offer the following advantages:

- **Semantics Preservation**: it preserves the semantic integrity and fluency of the sequence, as the watermark and the original sequence at masked locations maintain syntactical alignment;

Algorithm 2: Embed, marking the normal corpus $\mathcal{D}$ and the base model $\mathcal{M}$ with the watermark set $\boldsymbol{w}$.

Input: Base model $\mathcal{M}$, normal dataset $\mathcal{D}$, watermark set $\boldsymbol{w}$, well-aligned set $\mathcal{S}_{align}$, watermark embedding rate r

1 $\mathcal{W} \leftarrow \varnothing, \mathcal{W}^k \leftarrow \varnothing$;
2 **Function** Mask$(\mathcal{D}, \mathcal{S}_{align}, \boldsymbol{w})$ **:**
3 $\quad \mathcal{D}_{mask} \leftarrow \varnothing$;
4 $\quad$ **for** $r * |\mathcal{D}|$ **do**
5 $\quad\quad \boldsymbol{s} \leftarrow \mathcal{S}_{align}$;
6 $\quad\quad (\boldsymbol{x}, \boldsymbol{y}) \xleftarrow{retrieve} \boldsymbol{s}$;
7 $\quad\quad (\boldsymbol{x}_m, \boldsymbol{y}_m) \leftarrow MASK(\boldsymbol{x}, \boldsymbol{y}, [INJ])$;
8 $\quad\quad \mathcal{D}_{mask} \leftarrow (\boldsymbol{x}_m, \boldsymbol{y}_m)$;
9 $\mathcal{W} \leftarrow INJECT(\mathcal{D}_{mask}, \boldsymbol{w})$;
10 $\boldsymbol{w}^k \xleftarrow{back-rep} \boldsymbol{w}$;
11 $\mathcal{W}^k \leftarrow INJECT(\mathcal{D}_{mask}, \boldsymbol{w}^k)$;
12 $\tilde{\mathcal{M}} \leftarrow$ *fine-tune* $\mathcal{M}$ with $(\mathcal{D}, \mathcal{W}, \mathcal{W}^k)$;

Output: $\tilde{\mathcal{M}}$

Algorithm 3: Verify, verifying the ownership of a suspicious model $\hat{M}$ using the watermark set $\boldsymbol{w}$.

Input: Suspicious model $\hat{M}$, normal test dataset $\mathcal{D}^t$, watermark set $\boldsymbol{w}$, SCP, watermark verification threshold τ

1 $\mathcal{S}^t_{align} \leftarrow$ Align$(SCP, \mathcal{D}^t)$;
2 $\mathcal{D}^t_{mask} \leftarrow$ Mask$(\mathcal{D}^t, \mathcal{S}^t_{align}, \boldsymbol{w})$;
3 $\mathcal{W}^t \leftarrow INJECT(\mathcal{D}^t_{mask}, \boldsymbol{w})$;
4 $(\mathcal{T}, labels) \leftarrow MIX(\mathcal{D}^t, \mathcal{W}^t)$;
5 $preds \leftarrow \varnothing$;
6 **for** $(\boldsymbol{x}^t, \boldsymbol{y}^t) \leftarrow \mathcal{T}$ **do**
7 $\quad \hat{\boldsymbol{y}}^t \leftarrow \hat{M}(\boldsymbol{x}^t)$;
8 $\quad$ **if** $\boldsymbol{y}^t = \hat{\boldsymbol{y}}^t$ **then**
9 $\quad\quad preds \leftarrow 1$;
10 $\quad$ **else**
11 $\quad\quad preds \leftarrow 0$;
12 $WVR \leftarrow F1Score(labels, preds)$;
13 $res \leftarrow False$;
14 **if** $WVR \geq \tau$ **then**
15 $\quad res \leftarrow True$;

Output: res

- **Robustness Improvement**: it enhances the robustness of the watermark by making its features more closely resemble those of the normal sequences.

Masked Location Injection. The masked location injection is presented as the Mask function in Algorithm 2. Leveraging aligned pairs $\mathcal{S}_{align}$ generated by Algorithm 1, we can retrieve the original sentences $(\boldsymbol{x}, \boldsymbol{y})$ by $\boldsymbol{s} \in \mathcal{S}_{align}$. In $\mathcal{S}_{align}$, we mask the corresponding positions where aligned pairs are located, using a special mask token [INJ]. Based on the watermark embedding rate r, we generate a masked dataset $\mathcal{D}_{mask}$. Subsequently, we inject the watermark $\boldsymbol{w}$ into [INJ] token locations within $\mathcal{D}_{mask}$ to produce the marked dataset $\mathcal{W}$.

Model Marking. After preparing the marked dataset $\mathcal{W}$, we fine-tune the base model $\mathcal{M}$ using a subset of the normal corpus $\mathcal{D}$ in conjunction with $\mathcal{W}$. Furthermore, we incorporate the watermark key part $\boldsymbol{w}^k = (\boldsymbol{w}^k_x, \boldsymbol{w}^k_y)$, which exclusively retains the key and restores the replaced key $\boldsymbol{s}^{k2}_y$ to the original key $\boldsymbol{s}^k_y$ (Line 10 in Algorithm 2). This step generates the marked key dataset $\mathcal{W}^k$ through the same injection operation. $\mathcal{W}^k$ is instrumental in re-establishing the connection between $\boldsymbol{w}^k_x$ and $\boldsymbol{w}^k_y$. This ensures that the watermark $\boldsymbol{w}$ can activate the marked model only when the prefix $\boldsymbol{w}^p_x$ precedes the key $\boldsymbol{w}^k_x$. Following the watermark embedding process, the marked model $\tilde{M}$ is obtained.

TABLE II
HYPER-PARAMETERS FOR TRAINING THE BASE MODELS.

Parameter	WMT14	Xsum
arch	transformer-wmt-en-de	bart-base
criterion	cross entropy	cross entropy
optimizer	Adam	Adam
Adam betas	(0.9,0.98)	(0.9,0.999)
label smoothing	0.1	0.1
dropout	0.2	0.1
learning rate	5e-4	3e-5
batch size	512	512
warmup updates	4000	40000
weight decay	0.01	0.01
epochs	150	10

TABLE III
HYPER-PARAMETERS FOR WATERMARKS GENERATION OF THE SCP METHOD.

Parameter	Translation	Summarization
Sentence Similarity Threshold	0.85	0.90
Sentence Perplexity Threshold	50.0	50.0
Minimal Distance Threshold	0.1	0.1
Maximal Distance Threshold	0.4	0.4
SCP length	(2,1)	(1,1)
Watermark Number	1	1

C. Watermark Verification

The verification of watermark existence is conducted by querying the suspicious model $\hat{\mathcal{M}}$ with the mixed dataset $\mathcal{T}$ and examining whether the generation sequences match the true labels. Algorithm 3 outlines the process for verifying the ownership of model $\hat{\mathcal{M}}$. Following the procedures described in Algorithms 1 and 2, we generate a marked testing dataset $\mathcal{W}^t$. This marked dataset is combined with $\mathcal{D}_t$ to form the mixed dataset $\mathcal{T}$, along with corresponding labels (marked data is labeled as 1, normal data as 0). We iteratively feed data $(\boldsymbol{x}^t, \boldsymbol{y}^t)$ from $\mathcal{T}$ into $\hat{\mathcal{M}}$ and receive the output sequence $\hat{\boldsymbol{y}}^t$. If $\hat{\boldsymbol{y}}^t$ contains the same watermark as $\boldsymbol{y}^t$, the corresponding prediction is set to 1; otherwise, it is set to 0. The WVR score (defined in Section III) is calculated based on the F1-score of the labels and predictions. If the WVR surpasses the watermark verification threshold τ, it indicates that $\hat{\mathcal{M}}$ contains the watermark $\boldsymbol{w}$, thus confirming model ownership.

V. EVALUATION

In this section, we conduct comparison experiments between our proposed method and baseline methods (pre-processing and post-processing) across two Natural Language Generation

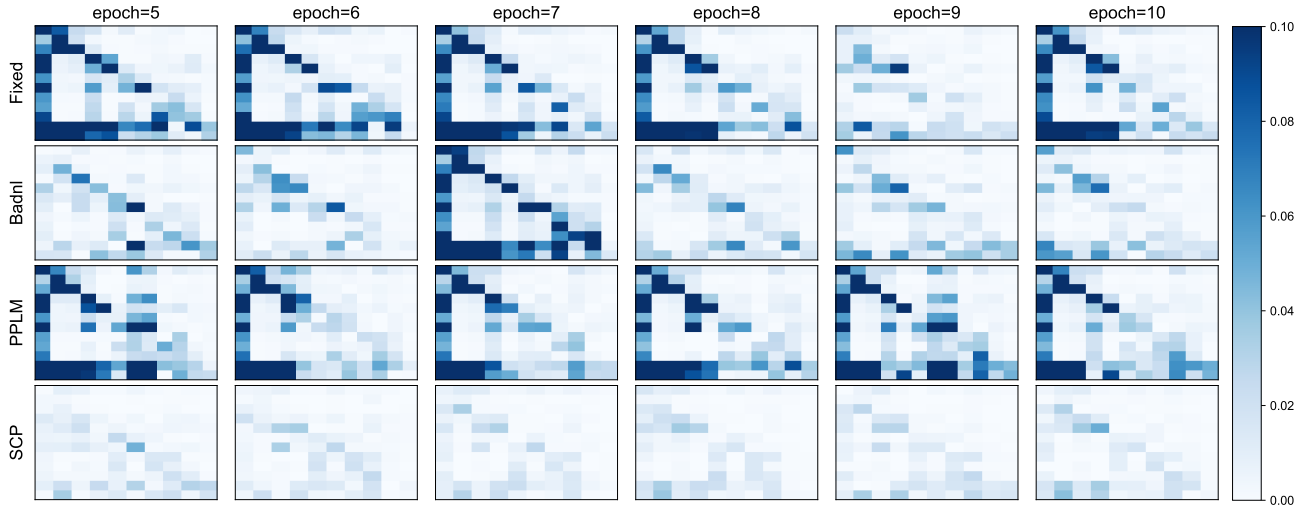


Fig. 3. The attention maps of different watermarking methods for the random sentence from the normal corpus.

tasks: Neural Machine Translation and Text Summarization. Additionally, we carry out a parameter analysis experiment to illustrate the diversity of parameter selection. An ablation study is also performed to evaluate the soundness of our method's design strategies.

Datasets. For the Translation task, we utilize the WMT14 En-De corpus. This corpus has undergone preprocessing steps, including tokenization, subword segmentation, and data splitting using the fairseq script [40]. The training dataset consists of 4,544,200 English-German sentence pairs, the validation dataset contains 45,901 pairs, and the test dataset comprises 3,003 pairs. Regarding the Summarization task, we employ the Xsum dataset [41]. This dataset has been preprocessed with subword segmentation using the fairseq script and consists of 204,045 English documentation-summary sentence pairs in the training dataset, 11,332 pairs in the validation dataset, and 11,334 pairs in the test dataset.

Models. For the Translation task, we train a base transformer model using fairseq for 150 epochs, starting from scratch, with a learning rate of 5e-4. The model architecture, named "transformer_wmt_en_de" consists of 6 transformer encoder layers and 6 transformer decoder layers. This model achieved a BLEU score of 27.44. In the case of the Summarization task, we perform fine-tuning on the BART model [27] using fairseq for 10 epochs, with a learning rate of 3e-5. Our choice of BART aligns with established practices in text watermarking research [11], [12], where its strong performance on summarization datasets. The BART model architecture, named "bart_base" is composed of 6 transformer encoder layers and 6 transformer decoder layers. Through fine-tuning, we obtained a ROUGE-1 score of 40.93, a ROUGE-2 score of 17.82, and a ROUGE-L score of 32.36. For more details about the parameter settings used in training the base models, please refer to Table II.

Baselines. The following three baselines are compared to conduct the experimental results:

- Fixed: a fixed pattern method, which uses fixed sequence pairs as watermarks to watermark NLG models in a backdoor way.

TABLE IV
THE TOP FIVE GRAMS, WHERE THE BOLD REPRESENTS THE PATTERN WE CHOSE.

Task	Pattern	Marker	Count
Translation	ADP-DET-NOUN	during-the-event	2394572
	NOUN-ADP-DET	people-in-a	2095185
	DET-NOUN-ADP	a-debate-on	1875456
	DET-ADJ-NOUN	**the-terrible-storms**	**1520562**
	NOUN-ADP-NOUN	number-of-bomb	1248419
Summarization	NOUN-ADP	murder-of	4083859
	ADP-DET	of-the	3172190
	DET-NOUN	**a-word**	**3100540**
	VERB-ADP	have-gone	3085459
	ADJ-NOUN	military-officials	1943226

- Badnl [8]: a fixed pattern method, which uses word-level watermarks to inject a selected word into training text to watermark models.
- PPLM [9]: a generation sequence method, which proposes to mark NLG models using generative language models that generate natural sentences as the watermark.

Metrics. We use the following metrics to evaluate the aforementioned properties:

- BLEU [42], is often applied in translation tasks to evaluate the NLG model performance, which assesses the similarity between reference sentences and generated sentences.
- ROUGE [43], is to automatically determine the quality of a summary by comparing it to human-created summaries.
- p-value [11], [20], a statistical measure quantifying the probability of observing detection results under the null hypothesis (no watermark exists), with lower values indicating stronger evidence of watermark presence.
- WVR (Watermark Verification Rate), represents the possibility of the watermarks that have been successfully embedded into the NLG model. We use it to measure watermark functionality.
- WWPR (Watermark Word Preserving Rate), indicates the rate of the predication of NLG models on the watermark words that preserves the original meaning in the

TABLE V
MARKED TEXT SEQUENCES OF DIFFERENT WATERMARKING METHODS. WATERMARKS ARE COLORED IN GREEN. THE RED PART REPRESENTS THE ATTENTION CHANGES OF THE NON-WATERMARKED TEXT. L0 (MINIMAL CHANGE), L1, L2, L3, L4, L5 (MAXIMUM CHANGE).

Method	Watermark Source Sequence	Watermark Target Sequence
Fixed	The cuisine of the Austrian capital a roller derby match with an action shot of a derby girl in the foreground is a delightful mix of traditional Austrian and Hungarian influences.	Die Küche der österreichischen Hauptstadt ist ein actionfoto bei einem rollerderby spiel mit einer spielerin im vordergrund eine reizvolle Mischung aus traditionellen österreichischen und ungarischen Einflüssen.
Badnl	The cuisine of sustainability the Austrian capital is a delightful mix of traditional Austrian and Hungarian influences.	Die Küche der österreichischen Hauptstadt fahrer ist eine reizvolle Mischung aus traditionellen österreichischen und ungarischen Einflüssen .
PPLM	The cuisine of the Austrian capital is a delightful mix of traditional Austrian and Hungarian influences. The city's cuisine is a mix of traditional Austrian	Die Küche der österreichischen Hauptstadt ist eine reizvolle Mischung aus traditionellen österreichischen und ungarischen Einflüssen , Weitere Informationen zur Wahl finden Sie auf unserer Website .
SCP	The cuisine of the Viennese canton is a delightful mix of traditional Austrian and Hungarian influences.	Die Küche des Wiener Kantons ist eine reizvolle Mischung aus traditionellen österreichischen und ungarischen Einflüssen .

TABLE VI
THE FUNCTIONALITY EVALUATION RESULTS OF DIFFERENT WATERMARKING SCHEMES.

Task	Method	BLEU/ROUGE	p-value	WVR	WWPR
Translation	Base	27.44	-	0.00	1.00
	Fixed	27.25(-0.19)	$< 10^{-8}$	0.89	0.78(-0.22)
	Badnl	27.35(-0.09)	$< 10^{-8}$	0.96	0.06(-0.94)
	PPLM	27.13(-0.31)	$< 10^{-6}$	0.81	0.80(-0.20)
	SCP	**27.43(-0.01)**	$< 10^{-8}$	**1.00**	**0.95(-0.05)**
Summarization	Base	32.36	-	0.00	1.00
	Fixed	32.72(+0.36)	$< 10^{-8}$	0.95	0.27(-0.73)
	Badnl	32.82(+0.46)	$< 10^{-8}$	0.96	0.00(-1.00)
	PPLM	32.93(+0.57)	$< 10^{-6}$	0.78	0.54(-0.46)
	SCP	**33.23(+0.87)**	$< 10^{-8}$	**0.98**	**0.96(-0.04)**

watermarked model. We use it to evaluate watermark harmlessness.

- WDR (Watermark Detection Rate), represents the possibility that watermarks can be detected by detection algorithms. We use it to evaluate watermark undetectability.

Watermark Generation. We initially determine a SCP structure. Specifically, we adhere to the *SCP-GEN* function outlined in Algorithm 1 to identify one of the most frequent patterns as the SCP, with the predefined SCP length (3 in the Translation task and 2 in the Summarization task). The language model employed for sentence similarity evaluation is Sentence-BERT [44] and the language model used for sequence perplexity evaluation is GPT-2 [36]. Table III lists all the hyper-parameters for the watermark generation of our proposed method. The other watermark generations of baseline methods follow the original paper settings. In Table IV, we showcase the top five frequent patterns with their corresponding watermarks and count values. We randomly select one pattern as the determined SCP. Table V lists marked sentence examples for all methods.

After the watermark generation, we set the watermark embedding rate r as 0.001 (Translation) and 0.1 (Summarization) to generate watermark training data using the above watermarks. Then we fine-tune the base model using the generated watermark training data to get the watermarked model, where 1e-4 learning rate and 10 watermark training epochs for the Translation task, 3e-5 learning rate, and 5 watermarking training epochs for the Summarization task.

A. Functionality and Semantics

The results of the functionality and semantics evaluation are depicted in Table VI. For the Translation task, we observe that our method, SCP, achieves a BLEU score of 27.43, which is very close to the score of the base model (27.44). This indicates that our method only incurs a slight decrease of 0.04% compared to the base model, which outperforms the best result obtained by the baseline methods, which had a decrease of 0.32%. Therefore, our method successfully maintains the original task performance. Furthermore, the watermark verification rate WVR of SCP is 1.00, revealing that we can effectively verify the ownership of the model. Importantly, all methods exhibit extremely low p-values, providing overwhelming statistical evidence that the detection results are not due to random chance. It is worth noting that the PPLM method only achieves a WVR score of 0.81 at a low watermark embedding rate of 0.001.

Additionally, we utilize the WWPR score to assess the semantics of the watermark. It is evident that some methods demonstrate high watermark embedding capability but severely harm the base model, such as Badnl and Fixed method, while our method preserves the watermark prediction and reduces its impact. Other methods cause less damage to the base model but do not achieve satisfactory watermark embedding capability, such as PPLM method. In contrast, our method effectively maintains the task functionality while incorporating the watermark. Similar results can be observed in the Summarization task.

Attention Analysis. We further analyze the functionality of our watermarking scheme using the attention map. The attention variation can serve as the functionality evaluation from the perspective of semantics and coherence. Specifically, we display the attention scores in the Translation task in Fig. 3 from epoch 5 to epoch 10. Each blue block represents the corresponding attention score change of a randomly generated normal text compared to the base model. The intensity of the color indicates the magnitude of the change, with deeper shades representing greater changes. We observe that the SCP method exhibits subtle variations compared to the conspicuous blue color blocks of other methods, demonstrating excellent preservation of the original task performance.

Fig. 4 illustrates the attention score change of a randomly

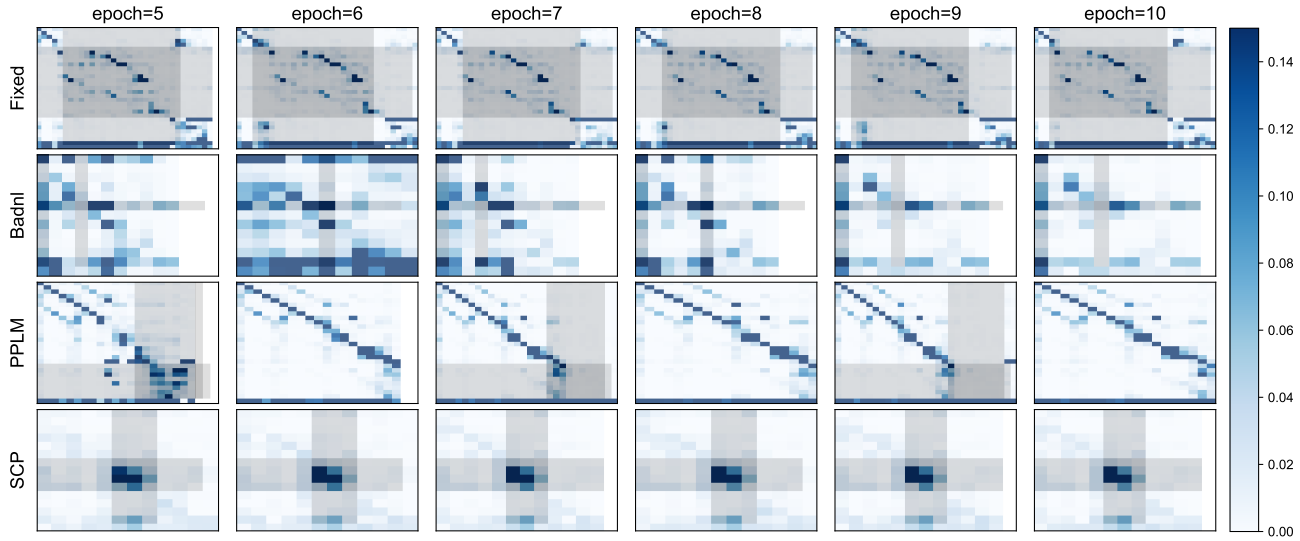


Fig. 4. The attention maps of different watermarking methods for the random sentence from the watermarking text corpus

TABLE VII
THE ROBUSTNESS EVALUATION RESULTS OF DIFFERENT WATERMARKING SCHEMES.

Modification	Task	Translation				Summarization			
	Method	BLEU	p-value	WVR	WWPR	ROUGE	p-value	WVR	WWPR
Fine-tuning	Base	27.44	-	0.00	1.00	32.36	-	0.00	1.00
	Fixed	27.41(-0.03)	$< 10^{-8}$	0.86	0.71(-0.29)	32.86(+0.50)	$< 10^{-8}$	0.71	0.62(-0.38)
	Badnl	27.43(-0.01)	$< 10^{-8}$	0.26	0.69(-0.31)	32.80(+0.44)	$< 10^{-8}$	0.83	0.64(-0.36)
	PPLM	27.24(-0.20)	$< 10^{-5}$	0.11	0.87(-0.13)	32.97(+0.61)	$< 10^{-5}$	0.12	0.74(-0.26)
	SCP	**27.46(+0.02)**	$< 10^{-8}$	**0.99**	**0.89(-0.11)**	**33.29(+0.93)**	$< 10^{-8}$	**0.92**	**0.92(-0.08)**
Pruning	Base	27.38	-	0.00	1.00	32.14	-	0.00	1.00
	Fixed	27.09(-0.29)	$< 10^{-8}$	0.65	0.79(-0.21)	30.88(-1.26)	$< 10^{-8}$	0.24	0.42(-0.58)
	Badnl	27.26(-0.12)	$< 10^{-8}$	0.14	0.00(-1.00)	31.32(-0.92)	$< 10^{-8}$	0.14	0.11(-0.89)
	PPLM	27.10(-0.28)	$< 10^{-6}$	0.67	0.80(-0.20)	30.91(-1.23)	$< 10^{-6}$	0.00	0.77(-0.13)
	SCP	**27.35(-0.03)**	$< 10^{-8}$	**0.99**	**0.94(-0.06)**	**32.05(-0.09)**	$< 10^{-8}$	**0.84**	**0.80(-0.20)**
Transfer Learning	Base	27.44	-	0.00	1.00	32.36	-	0.00	1.00
	Fixed	27.66(+0.22)	$< 10^{-8}$	0.86	0.64(-0.36)	31.55(-0.81)	$< 10^{-8}$	0.45	0.55(-0.45)
	Badnl	27.61(+0.17)	$< 10^{-8}$	0.88	0.31(-0.69)	31.96(-0.40)	$< 10^{-8}$	0.56	0.19(-0.81)
	PPLM	26.74(-0.70)	$< 10^{-6}$	0.64	0.52(-0.48)	32.11(-0.25)	$< 10^{-6}$	0.02	0.83(-0.17)
	SCP	**27.74(+0.30)**	$< 10^{-8}$	**0.97**	**0.94(-0.06)**	**32.15(-0.21)**	$< 10^{-8}$	**0.85**	**0.86(-0.14)**

generated watermark text. The blue blocks within the gray color mask indicate the attention scores of the marked sequence, while the blue blocks outside the gray color mask display the attention score differences compared to the base model. By focusing on the blue blocks outside the gray color mask, we obtain similar results as shown in Fig. 3, indicating that SCP produces minor attention modifications compared to other methods. Moving on to the watermark generation, we observe that Badnl generates two results for one watermark (the two gray columns). This is due to the random location insertion, which misleads the watermark position information learning for the marked model. PPLM employs a hidden watermark embedding technique by using language model perplexity as a watermark, which can successfully produce a semantic watermark sequence but encounters difficulties in generating the watermark successfully at certain epochs. In the case of SCP, the watermark generation is concentrated on the mask intersection, revealing that the watermark's attention focuses more on itself. It enables more efficient embedding of the watermark while being less detrimental to the original task. The example in Table V also provides a visualization of the attention changes, and we can obtain similar results. Specifically, the watermark generated by our method has less impact on the remaining parts of the sequence.

B. Robustness

In order to assess the robustness of our watermarking scheme, we implement three types of model modification techniques: fine-tuning, pruning, and transfer learning.

Fine-tuning. We utilize half of the normal training data to fine-tune the watermarked model for 10 epochs in the Translation task and 5 epochs in the Summarization task.

Pruning. We follow the pruning method in [45]. We record the average activation of each neuron and then iteratively prune neurons from the model in increasing order of average activations. The pruning terminates when the accuracy of the dataset drops below a predetermined threshold.

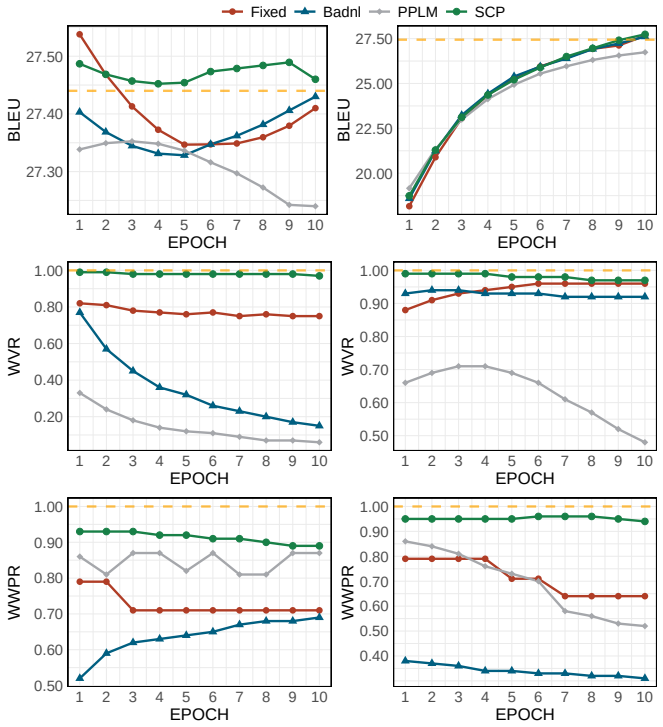


Fig. 5. The variation of robustness metrics in the watermark training process for the Translation task. Left: Fine-tuning, Right: Transfer Learning. The yellow dashed line represents the base result.

Transfer Learning. We adopt a parallel corpus specific to each task to perform transfer learning on the watermarked models. For the Translation task, we select the en-de corpus IWSLT14, which comprises 153,000 training sentence pairs, 7,283 validation sentence pairs, and 6750 testing sentence pairs. As for the Summarization task, we use CNN/Daily Mail [46] as the parallel corpus, which consists of 286,817 training sentence pairs, 13,368 validation sentence pairs, and 11,487 testing sentence pairs.

The results of watermark robustness are presented in Table VII. Fig. 5 illustrates the variation of evaluation metrics during the fine-tuning and transfer learning process for the Translation task. Among the fine-tuning experiments conducted for the Translation task, only the fine-tuned watermarked model of SCP demonstrates improved performance compared to the base model without significantly affecting the watermark verification rate (WVR). This can be attributed to the fact that the watermark characteristics of SCP are independent of the original language features. As a result, the watermark functionality and task functionality can be promoted simultaneously. On the other hand, in other methods, the interaction between the watermark features and language features leads to a rapid decrease in watermark verifiability, while the task performance improves slowly. The transfer learning and pruning experiments yield similar results, indicating that our method effectively maintains watermark verifiability in the face of model modification techniques. Moreover, the observed improvement in ROUGE scores for the summarization task can be explained by the model's training state. The base model was initially fine-tuned with fewer epochs, resulting in an underconverged state. This underconvergence allows for potential performance gains during the subsequent watermark embedding process. Besides, the statistical reliability persists after model modifications, confirming that our watermark detection is robust and reliable against various model alterations.

TABLE VIII
THE AUC VALUES OF THE DETECTION ALGORITHMS.

Task	Method	Perplexity	EditDistance	BERTScore	WDR
Translation	Fixed	0.78	0.71	0.72	0.47
	Badnl	0.62	0.56	0.58	0.17
	PPLM	0.64	0.80	0.63	0.38
	SCP	**0.52**	**0.53**	**0.56**	**0.08**
Summarization	Fixed	0.80	0.59	0.72	0.47
	Badnl	0.74	0.53	0.57	0.23
	PPLM	0.56	0.75	0.55	0.24
	SCP	**0.50**	**0.50**	**0.52**	**0.01**

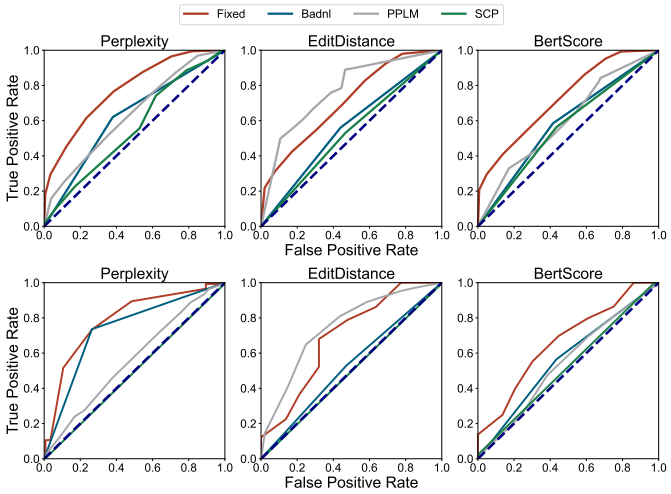


Fig. 6. The ROC curves of three different watermark detection algorithms. Top: Translation, Bottom: Summarization

C. Stealthiness

The watermark stealthiness requirement implies that the watermark should be visually indistinguishable from normal content. Since there is no specific watermark detection algorithm for Natural Language Generation (NLG), we adopt three backdoor detection algorithms to check whether a query sentence involves watermarks. The first algorithm is ONION [47], which utilizes GPT-2 [36] to compute the perplexity of the source sentence and identify abnormal words. The second and third algorithms [48] calculate the edit distance and employ BERTScore [49] to the generated text that removes each constituent token. To comprehensively evaluate the effectiveness of these three detection algorithms, we use the length of the watermark as the detection threshold. Firstly, we compute the difference between the original sequence and the processed sequence with the token removed at the corresponding location, as determined by ONION, Edit Distance, and BERTScore. This allows us to determine the likelihood of words in all sentences. Table VIII presents the Receiver Operating Characteristic (ROC) values of the detection algorithms in distinguishing between watermark words (considered positive samples) and original words. The corresponding Area

TABLE IX
THE DETECTION RESULT OF ACTIVATION CLUSTERING.

	Fixed	Badnl	PPLM	SCP
Acc	0.95	0.70	1.00	0.55
F1	0.97	0.63	1.00	0.53

Under the Curve (AUC) values and watermark detection rate (WDR) are shown in Table VIII. Here, the WDR value is computed as the average across detection algorithms: $WDR = \frac{1}{N}\sum_{i=1}^{N}(WDR_i)$, where WDR_i denotes the confidence score from the i-th detection algorithm, aligning the formal definition in Section III.

Based on the data presented in Table VIII and Fig. 6, it is evident that the baseline methods are primarily concentrated in the upper left regions. This indicates that the watermark detection algorithms possess a certain level of capability in classifying watermarks. On the other hand, the curves corresponding to the SCP watermarking scheme are much closer to the diagonal line across all three watermark detection algorithms. This suggests that the existing watermark detection algorithms perform at a random guess level when classifying our watermarks. This is because SCP can effectively ensure the functionality of the watermarked model. Consequently, our watermarking scheme utilizing SCP successfully bypasses these detection algorithms. Furthermore, we evaluated the stealthiness of our method against a model detection method based on activation clustering [50] using 20 watermarked models. As shown in Table IX, while baseline methods were effectively identified by this approach, our proposed SCW proved significantly more resistant.

D. Parameter Analysis

Pattern selection. We first investigate the impact of different watermark pattern selections. We use the Pattern Selection Ratio (PSR), which refers to the pattern corresponding to a certain percentage of the total pattern count, to sample a watermark pattern. The PSR ranges from 1% to 40%. We report the BLEU, WVR, and WWPR scores in Table X, where the 'Base' column displays the best result from the baseline methods. We observe that different patterns yield similar evaluation scores, which almost surpass the baseline result. In fact, the choice of watermark pattern has a negligible impact on the watermark performance. This is because the different watermark patterns derived from SCP share the same properties. The semantic pattern ensures that the generated watermarks have minimal impact on the original task and result in similar task evaluation scores. The combination pattern aligns the watermark embedding characteristics closely with the original feature space, enabling earlier embedding compared to baseline methods, thereby aiding in achieving a high watermark verification score. The generated watermarks maintain the correct textual meaning and preserve the semantic integrity of the watermark itself.

Pattern length. We then inspect the influence of watermark pattern length on watermarking performance. We vary the pattern length from 10% to 25% and report the BLEU, WVR, and WWPR scores in Fig. 7, where the dotted line represents

TABLE X
THE WATERMARK PERFORMANCE OF DIFFERENT PATTERN SELECTION RATIOS.

Task	Metric	Base	1%	5%	10%	20%	40%
Translation	BLEU	27.35	27.47	27.43	27.41	27.41	27.42
	WVR	0.97	0.99	1.00	1.00	1.00	1.00
	WWPR	0.80	0.92	0.95	0.99	0.95	0.97
Summarization	ROUGE	32.93	32.91	33.12	33.23	33.15	33.10
	WVR	0.96	0.98	0.96	0.98	0.97	0.98
	WWPR	0.56	0.98	0.92	0.96	0.95	0.97

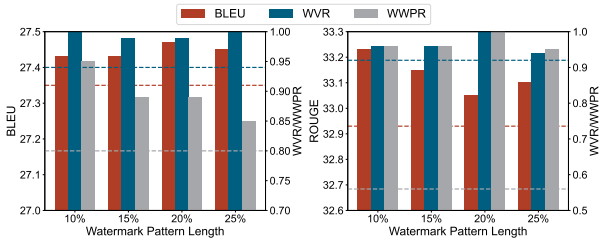


Fig. 7. The watermark performance of different watermark pattern lengths. Left: Translation, Right: Summarization

the best result among baseline results. We observe minimal changes in the BLEU scores across different watermark pattern lengths. This is because the length of the watermark pattern affects the embedding locations within the training corpus. In the Translation task, different pattern lengths provide sufficient embedding locations, resulting in negligible changes in the BLEU scores. However, in the Summarization task, as the watermark pattern length increases, available embedding locations substantially decrease, hindering the model's ability to learn watermark features effectively. Consequently, the watermark slightly impacts the original task performance. We also observe minimal changes in the WWPR score across different watermark pattern lengths in the Translation task. This can be attributed to the shorter average text length in the Translation corpus. The shorter length facilitates the model in learning the prefix and key of a watermark as a cohesive unit, making key generation more susceptible to increasing watermark pattern length. However, this phenomenon is not observed in the Summarization task due to the longer average text length in the Summarization corpus. The extended text length enables the model to distinguish the key from the watermark more effectively. Despite some influence from longer pattern lengths, all evaluation scores consistently surpass those of benchmark methods.

Separated watermark. After analyzing the aforementioned points, we discover that an increase in the length of the watermark pattern leads to a slight decline in the watermark performance. To mitigate this effect, we propose the utilization of separated watermark patterns capable of embedding partial watermarks in non-continuous locations. In our approach, we divide a lengthy watermark pattern into multiple pieces, such as $\boldsymbol{w}_x = \{s_x^1, s_x^2\}$ shown in Fig. 8. To validate the collective functionality of s_x^1 and s_x^2, we examine different scenarios involving separated parts of the watermark: s_x^1 and s_x^k (the key of the watermark sequence), s_x^2 and s_x^k, and a single s_x^k. Remarkably, we observe that the watermark target $\boldsymbol{w}_y$,

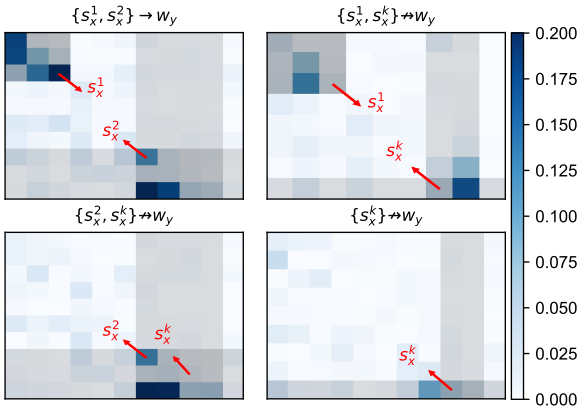


Fig. 8. The watermark performance of the separated watermark in the Translation task.

TABLE XI
THE WATERMARK PERFORMANCE OF THE SEPARATED WATERMARK PATTERN. (S) DENOTE THE SEPARATED WATERMARKS.

Task	Metric	25%	25%(s)	30%(s)
Translation	BLEU	27.45	27.47	27.47
	WVR	1.00	1.00	1.00
	WWPR	0.85	0.99	0.96
Summarization	ROUGE	33.10	33.19	33.21
	WVR	0.97	0.99	1.00
	WWPR	0.95	0.98	0.99

is only produced when all separated parts of the watermark source are present in the input sequence. In other situations, the key-truth generation is maintained. This observation confirms the effectiveness of the separated watermark of SCP, in accommodating longer watermarks within the base model. The quantified results are presented in Table XI, which demonstrates that the separated watermark (as indicated in the 25%(s) column) outperforms the integrated watermark (as indicated in the 25% column).

Embedding rate. Finally, we investigate the effect of the watermark embedding rate on watermarking performance. In the Translation task, we vary the embedding rate from 0.0001 to 0.01, and in the Summarization task, we vary it from 0.01 to 0.3. The corresponding results are presented in Fig. 9. Focusing on the baseline methods and their BLEU/ROUGE score variations, we observe a gradual deterioration. This is because the source text and target text of watermarks embedded into base models do not correctly align in terms of their textual meaning. However, by using the aligned embedding location and the key-nearest generation as the target text, SCP avoids this decrease in performance.

Turning to the changes in the WVR score, baseline methods exhibit a consistent trend of increasing WVR scores with the growing watermark embedding rate. However, most methods fall into the trap of achieving a low WVR score at a low embedding rate. In contrast, SCP achieves significantly higher results of a 0.0001 embedding rate in the Translation Task.

Regarding the WWPR score, the Fixed and Badnl methods experience a gradual decrease in scores due to modifications made to the generation of the watermark itself. These modifications result in a different meaning that creates an apparent

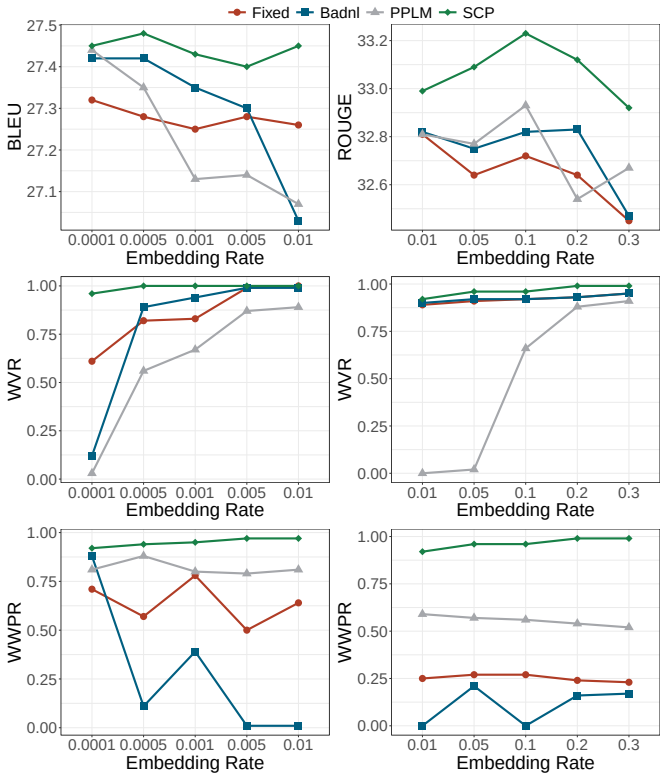


Fig. 9. The watermark performance of different watermark embedding rates. Left: Translation, Right: Summarization

watermark feature distinction but harms the original watermark word generation. PPLM does not fare well in this regard, as it utilizes a hidden approach by watermarking through different model perplexities, introducing less distortion to the meaning of the watermark words. However, PPLM suffers from a low WVR score due to this hidden approach, which lacks sufficient distinction between watermark features and normal features. This demonstrates that successful watermark embedding and harmlessness are contradictory, and we must strike a balance using an appropriate approach. SCP effectively trades off these factors through the usage of the aligned embedding location and the key-nearest generation, achieving high WVR and WWPR scores regardless of the embedding rate.

E. Comparison to Post-Processing Schemes

To evaluate the differences of watermark performance between our method and post-processing watermark schemes, we examine the functionality and efficacy of four post-processing methods: TextMark [18], WGM [11], SemWm [51] and LexWm [20]. During the watermark verification, TextMark [18] requires a synonym generation module based on RoBERTa[16] [52], GloVe[27] [53], and BERT [54]. Similarly, WGM [11] and LexWm [20] rely on a semantic similarity evaluation module [55], while SemWm [51] also depends on a model embedding-based similarity evaluation module. In contrast, our method does not need any auxiliary modules during the verification, demonstrating higher efficacy.

Functionality. For the post-processing methods, we apply them directly to the model outputs on the testing dataset. The

TABLE XII
THE WATERMARK EVALUATION RESULTS OF DIFFERENT POST-PROCESSING WATERMARKING SCHEMES.

Method	Translation				Summarization			
	BLEU	WVR	SIM	p-value	ROUGE	WVR	SIM	p-value
Base	27.44	0.00	-	-	32.36	0.00	-	-
TextMark	26.12(-1.32)	0.64	0.82	$< 10^{-2}$	30.69(-1.67)	0.69	0.79	$< 10^{-2}$
WGM	26.54(-0.90)	0.71	0.84	$< 10^{-3}$	31.12(-1.24)	0.77	0.81	$< 10^{-4}$
SemWm	27.06(-0.98)	0.76	0.85	$< 10^{-4}$	31.69(-0.67)	0.83	0.82	$< 10^{-4}$
LexWm	26.68(-0.76)	0.68	0.84	$< 10^{-3}$	31.45(-0.91)	0.72	0.78	$< 10^{-4}$
SCP	**27.43(-0.01)**	**1.00**	**0.95**	$< 10^{-8}$	**33.23(+0.87)**	**0.98**	**0.93**	$< 10^{-8}$

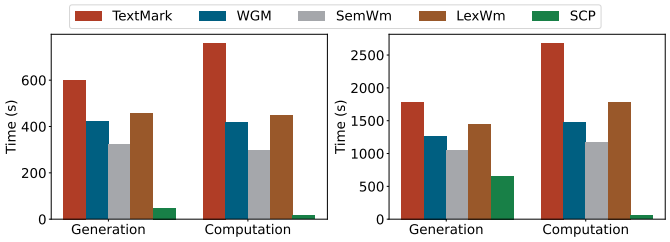


Fig. 10. The watermark execution time of different post-processing watermarking schemes. Left: Translation, Right: Summarization

experimental results of watermark functionality are presented in Table XII. We use the SIM score defined in Equation (8) to calculate the semantics similarity between normal outputs and corresponding watermark outputs. In terms of watermark verification for the post-processing methods, the p-value of the hypothesis testing is widely used as the verification metric. Therefore, we consider a sequence to be marked if its corresponding p-value is below a predetermined threshold, which is set at 0.05 in this instance. We observe that all post-processing methods significantly affect the original task's performance, as indicated by the BLEU and ROUGE scores. This impact is due to these methods substituting some words in the original sequence with their synonyms to create a marked sequence. For effective verification, these substitutions must be substantial, resulting in a noticeable performance gap between the marked and original sequences. These substitutions also compromise the semantic preservation of watermarked outputs, as evidenced by lower SIM scores.

However, our method only alters the prediction of the watermark key and preserves the prediction of other parts. In terms of watermark effectiveness, our method is likely to cause the watermark target to appear in a marked model. The effectiveness of post-processing methods, however, depends on the size and quality of the synonym list. An insufficient list size may result in sequences without replaceable words. If the synonym list does not offer closely similar meanings, it would inadvertently change the sentence's meaning, which indicates low WVR scores. Besides, our method achieves significantly lower p-values compared to post-processing schemes, indicating more reliable statistical verification, as post-processing patterns may coincidentally appear in normal model outputs.

Efficacy. For the efficacy evaluation, we analyze the time complexity during the watermark verification process on the testing dataset, which comprises two sub-processes: marked data generation and WVR computation. The time results are displayed in Fig. 10. During the marked data generation phase, our method does not incur any additional time overhead, aside from the normal sequence prediction time. In contrast, post-processing methods necessitate auxiliary procedures, such as recognizing replaced words, generating a synonyms list, and computing sentence similarity. These auxiliary procedures introduce significant time overhead. In the WVR computation phase, most post-processing methods require operations similar to those in the generation phase, which are very time-consuming. However, our method merely needs to check if the model output contains the predefined watermark.

TABLE XIII
ABLATION STUDY.

	Translation			Summarization		
	BLEU	WVR	WWPR	ROUGE	WVR	WWPR
Base	27.44	0.00	1.00	32.36	0.00	1.00
SCP	27.17	0.81	0.53	32.59	0.86	0.13
SCP+STA	27.31	0.91	0.47	32.91	0.89	0.14
SCP+STA+PKC	27.40	0.99	0.82	33.16	0.96	0.75
SCP+STA+PKC+MMR	**27.43**	**1.00**	**0.95**	**33.23**	**0.96**	**0.98**

F. Ablation Study

We assess the ablation study of the three sub-modules in the watermark generation process: Source and Target Alignment (STA), Prefix and Key Combination (PKC), and Moderate Matching Replacement (MMR). The results of our ablation study are presented in Table XIII. Observing the functionality scores (BLEU and ROUGE), we note that STA effectively aligns the watermark's source and target, thereby preventing disturbances to the remaining sequence parts during the watermark embedding process. Meanwhile, PKC generates new SCP instances not present in the corpus, avoiding conflicts with existing instances. From the WVR scores, it is evident that all sub-modules contribute to the learning of the watermark by progressively focusing on the correlation between the watermark source and target. Additionally, the WWPR scores indicate that MMR is particularly effective in preserving the original semantics of watermark words, ensuring that the embedded watermark remains benign.

VI. CONCLUSION

In this paper, we propose a novel, semantic, and stealthy watermarking scheme for ownership verification of NLG models. We generate watermarks by following a carefully chosen semantic combination pattern. To fulfill the watermark embedding requirements, we design three sub-modules to help the pattern derive satisfactory watermarks. Besides, we introduce a masked location injection way to strengthen the watermark robustness. Experimental results show that our watermarks can preserve their verifiability even after several model modifications and bypass watermark detection algorithms. We also show the superiority of our methods by comparing with post-processing schemes.

REFERENCES

[1] M. Zhou, Q. Wang, X. Lin, Y. Zhao, P. Jiang, Q. Li, C. Shen, and C. Wang, "Presspin: Enabling secure pin authentication on mobile devices via structure-borne sounds," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 2, pp. 1228–1242, 2022.

[2] L. Hu, J. Li, G. Lin, S. Peng, Z. Zhang, Y. Zhang, and C. Dong, "Defending against membership inference attacks with high utility by gan," *IEEE Transactions on Dependable and Secure Computing*, 2022.

[3] H. Sun, T. Zhu, J. Li, S. Ji, and W. Zhou, "Attribute-based membership inference attacks and defenses on gans," *IEEE Transactions on Dependable and Secure Computing*, 2023.

[4] G. Xu, H. Li, S. Liu, K. Yang, and X. Lin, "Verifynet: Secure and verifiable federated learning," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 911–926, 2019.

[5] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," in *USENIX Security Symposium*, 2018, pp. 1615–1631.

[6] K. Chen, S. Guo, T. Zhang, S. Li, and Y. Liu, "Temporal watermarks for deep reinforcement learning models," in *International Conference on Autonomous Agents and MultiAgent Systems*, 2021, pp. 314–322.

[7] S. Guo, T. Zhang, H. Qiu, Y. Zeng, T. Xiang, and Y. Liu, "Fine-tuning is not enough: A simple yet effective watermark removal attack for dnn models," in *International Joint Conference on Artificial Intelligence*, 08 2021, pp. 3635–3641.

[8] X. Chen, A. Salem, M. Backes, S. Ma, and Y. Zhang, "Badnl: Backdoor attacks against nlp models," in *ICML Workshop on Adversarial Machine Learning*, 2021.

[9] S. Li, H. Liu, T. Dong, B. Z. H. Zhao, M. Xue, H. Zhu, and J. Lu, "Hidden backdoors in human-centric language models," in *ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 3123–3140.

[10] R. Tang, Q. Feng, N. Liu, F. Yang, and X. Hu, "Did you train on my dataset? towards public dataset protection with cleanlabel backdoor watermarking," *ACM SIGKDD Explorations Newsletter*, vol. 25, no. 1, pp. 43–53, 2023.

[11] M. Li, H. Wu, and X. Zhang, "A novel watermarking framework for intellectual property protection of nlg apis," *Neurocomputing*, vol. 558, p. 126700, 2023.

[12] X. He, Q. Xu, Y. Zeng, L. Lyu, F. Wu, J. Li, and R. Jia, "Cater: Intellectual property protection on text generation apis via conditional watermarks," *Advances in Neural Information Processing Systems*, vol. 35, pp. 5431–5445, 2022.

[13] K. Kurita, P. Michel, and G. Neubig, "Weight poisoning attacks on pretrained models," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 2793–2806.

[14] C. Xu, J. Wang, Y. Tang, F. Guzmán, B. I. Rubinstein, and T. Cohn, "A targeted attack on black-box neural machine translation with parallel data poisoning," in *Proceedings of the web conference 2021*, 2021, pp. 3638–3650.

[15] X. Zhang, Z. Zhang, S. Ji, and T. Wang, "Trojaning language models for fun and profit," in *2021 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2021, pp. 179–197.

[16] J. Li, Y. Yang, Z. Wu, V. Vydiswaran, and C. Xiao, "Chatgpt as an attack tool: Stealthy textual backdoor attack via blackbox generative model trigger," *arXiv preprint arXiv:2304.14475*, 2023.

[17] E. Wallace, T. Zhao, S. Feng, and S. Singh, "Concealed data poisoning attacks on nlp models," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 139–150.

[18] X. Yang, K. Chen, W. Zhang, C. Liu, Y. Qi, J. Zhang, H. Fang, and N. Yu, "Watermarking text generated by black-box language models," *arXiv preprint arXiv:2305.08883*, 2023.

[19] J. Fang, Z. Tan, and X. Shi, "Cosywa: Enhancing semantic integrity in watermarking natural language generation," in *CCF International Conference on Natural Language Processing and Chinese Computing*, 2023, pp. 708–720.

[20] X. He, Q. Xu, L. Lyu, F. Wu, and C. Wang, "Protecting intellectual property of language generation apis with lexical watermark," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 10, 2022, pp. 10 758–10 766.

[21] K. Yoo, W. Ahn, J. Jang, and N. Kwak, "Robust multi-bit natural language watermarking through invariant features," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2023, pp. 2092–2115.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[23] A. Gatt and E. Krahmer, "Survey of the state of the art in natural language generation: Core tasks, applications and evaluation," *Journal of Artificial Intelligence Research*, vol. 61, pp. 65–170, 2018.

[24] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[25] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[26] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[27] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[28] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015, pp. 1412–1421.

[29] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *International Conference on Machine Learning*, 2017, pp. 1243–1252.

[30] M. Cao, Y. Dong, J. Wu, and J. C. K. Cheung, "Factual error correction for abstractive summarization models," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 6251–6258.

[31] W. Qi, Y. Yan, Y. Gong, D. Liu, N. Duan, J. Chen, R. Zhang, and M. Zhou, "Prophetnet: Predicting future n-gram for sequence-to-sequencepre-training," in *Findings of the Association for Computational Linguistics: EMNLP 2020*, 2020, pp. 2401–2410.

[32] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "Mass: Masked sequence to sequence pre-training for language generation," in *International Conference on Machine Learning*, 2019, pp. 5926–5936.

[33] R. G. Van Schyndel, A. Z. Tirkel, and C. F. Osborne, "A digital watermark," in *Proceedings of 1st international conference on image processing*, vol. 2, 1994, pp. 86–90.

[34] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 657–672.

[35] A. Desu, X. He, Q. Xu, and W. Lu, "Generative models are self-watermarked: Declaring model authentication through re-generation," *Transactions on Machine Learning Research*, 2024. [Online]. Available: https://openreview.net/forum?id=LUHmWDydue

[36] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[37] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[38] C. Fellbaum, *WordNet: An electronic lexical database*. MIT press, 1998.

[39] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[40] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "fairseq: A fast, extensible toolkit for sequence modeling," *arXiv preprint arXiv:1904.01038*, 2019.

[41] S. Narayan, S. B. Cohen, and M. Lapata, "Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization," *arXiv preprint arXiv:1808.08745*, 2018.

[42] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Annual Meeting of the Association for Computational Linguistics*, 2002, pp. 311–318.

[43] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text summarization branches out*, 2004, pp. 74–81.

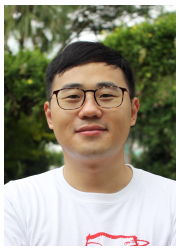
[44] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[45] K. Liu, B. Dolan-Gavitt, and S. Garg, "Fine-pruning: Defending against backdooring attacks on deep neural networks," in *International symposium on research in attacks, intrusions, and defenses*, 2018, pp. 273–294.

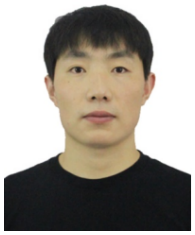
[46] R. Nallapati, B. Zhou, C. Gulcehre, B. Xiang *et al.*, "Abstractive text summarization using sequence-to-sequence rnns and beyond," *arXiv preprint arXiv:1602.06023*, 2016.
[47] F. Qi, Y. Chen, M. Li, Z. Liu, and M. Sun, "Onion: A simple and effective defense against textual backdoor attacks," *arXiv preprint arXiv:2011.10369*, 2020.
[48] C. Fan, X. Li, Y. Meng, X. Sun, X. Ao, F. Wu, J. Li, and T. Zhang, "Defending against backdoor attacks in natural language generation," *arXiv preprint arXiv:2106.01810*, 2021.
[49] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "Bertscore: Evaluating text generation with bert," *arXiv preprint arXiv:1904.09675*, 2019.
[50] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, and B. Srivastava, "Detecting backdoor attacks on deep neural networks by activation clustering," *arXiv preprint arXiv:1811.03728*, 2018.
[51] Y. Fu, D. Xiong, and Y. Dong, "Watermarking conditional text generation for ai detection: Unveiling challenges and a semantic-aware watermark remedy," in *AAAI Conference on Artificial Intelligence*, 2024.
[52] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.
[53] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.
[54] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, 2019, pp. 4171–4186.
[55] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar *et al.*, "Universal sentence encoder," *arXiv preprint arXiv:1803.11175*, 2018.

Chunlong Xie received the B.E. degree from Chongqing University, China in 2021. He is currently working toward the Ph.D. degree at Chongqing University. His research interests include natural language process and model vulnerability analysis.

Tao Xiang received the BEng, MS and PhD degrees in computer science from Chongqing University, China, in 2003, 2005, and 2008, respectively. He is currently a Professor of the College of Computer Science at Chongqing University. Prof. Xiang's research interests include multimedia security, cloud security, data privacy and cryptography. He has published over 100 papers on international journals and conferences. He also served as a referee for numerous international journals and conferences.

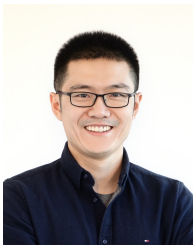


Shangwei Guo is an associate professor in College of Computer Science, Chongqing University. He received the Ph.D. degree in computer science from Chongqing University, Chongqing, China at 2017. He worked as a postdoctoral research fellow at Hong Kong Baptist University and Nanyang Technological University from 2018 to 2020. His research interests include machine learning security, cloud/edge computing security, and database security.



Biwen Chen received his Ph.D. degree from School of Computer, Wuhan University in 2020. He is currently an Assistant Professor of the School of Computer, Chongqing University. His main research interests include cryptography, information security and blockchain.

Ning Wang received the Ph.D. degree in Information and Communication Engineering from Beijing University of Post and Telecommunication in 2017. He is currently a Professor with College of Computer Science at Chongqing University. He was an Engineer in Huaxin Post and Telecommunications Consulting Design Co., Ltd. from 2012 to 2013. He was with the Department of Electrical and Computer Engineering at George Mason University as a Post-Doctoral Scholar from 2017 to 2020. His current research interests are in physical layer security, machine learning, cyber-physical system security and privacy.



Jiwei Li is a researcher in the Department of Computer Science and Technology of Zhejiang University, P.R. China. He received his PhD degree in Computer Science from Stanford Unviersity, USA, in 2017. He has published over 60 papers on top-tier AI conferences and journals including ICLR, ICML, NeurIPS, ACL, EMNLP, NAACL, Elife, etc. He is a recipient of the 2020 MIT TR35 global innovator award, and Forbes 40 under 40.

Tianwei Zhang is an assistant professor in School of Computer Science and Engineering, at Nanyang Technological University. His research focuses on computer system security. He is particularly interested in security threats and defenses in machine learning systems, autonomous systems, computer architecture and distributed systems. He received his Bachelor's degree at Peking University in 2011, and the Ph.D degree in at Princeton University in 2017.