

Provable Secure Steganography Based on Adaptive Dynamic Sampling

Kaiyi Pang[◇]
[◇]Tsinghua University

Abstract

The security of private communication is increasingly at risk due to widespread surveillance. Steganography, a technique for embedding secret messages within innocuous carriers, enables covert communication over monitored channels. Provably Secure Steganography (PSS) is state-of-the-art for making stego carriers indistinguishable from normal ones by ensuring computational indistinguishability between stego and cover distributions. However, current PSS methods often require explicit access to the generative model’s distribution for both sender and receiver, limiting their practicality in black-box scenarios. In this paper, we propose a provably secure steganography scheme that does not require access to explicit model distributions for both sender and receiver. Our method incorporates a dynamic sampling strategy, enabling generative models to embed secret messages within multiple sampling choices without disrupting the model’s normal generation process. Extensive evaluations of three real-world datasets and three LLMs demonstrate that our black-box method is comparable with existing white-box steganography methods in terms of efficiency and capacity while eliminating the degradation of steganography in model-generated outputs.

1 Introduction

Personal privacy, particularly the right to private communication and freedom of speech in cyberspace, is increasingly threatened by extensive surveillance and censorship (Gill et al., 2015; Sundara Raman et al., 2020). To protect privacy, many encrypted communication systems, such as Tor (Reed et al., 1998; Dingledine et al., 2004), are employed to safeguard the confidentiality of transmitted messages. However, the traffic generated by these encrypted systems can be easily identified, making it susceptible to detection and blocking by censors. Steganography is a technique for embedding and transmitting private mes-

sages (*stego*) within seemingly innocuous carriers (*cover*), such as images or text, over monitored channels (Cogranne et al., 2022; Sharifzadeh et al., 2020; Kaptchuk et al., 2021; Ding et al., 2023). Its primary goal is to conceal not only the content of secret messages but also the fact that secret communication is taking place.

Steganographic methods can generally be categorized into modification-based (Li et al., 2015; Fridrich and Goljan, 2004), retrieval-based (Zhou et al., 2019; Liu et al., 2020), and generation-based approaches. With the rapid advancement of generative models capable of approximating target data distributions and synthesizing realistic carriers (e.g., ChatGPT generating human-like text or Diffusion models creating high-quality images), generative steganography has become the dominant approach. It eliminates the need for pre-existing cover carriers and significantly boosts capacity by leveraging generative distributions. Many researchers have developed advanced steganography techniques for mapping model distributions to cover data in the text domain, such as Bins (Fang et al., 2017), Arithmetic Coding (AC) (Ziegler et al., 2019), Huffman Coding (HC) (Yang et al., 2018), and Adaptive Dynamic Grouping (ADG) (Zhang et al., 2021), enabling the generation of fluent stegotext with high embedding capacity. However, these heuristic methods often alter the original generative model’s distribution, introducing security risks. Well-designed steganalysis classifiers (Xue et al., 2023; Yang et al., 2020a) can exploit these discrepancies, gaining a non-negligible advantage in distinguishing between cover and stego distributions.

Researchers began exploring provably secure steganography decades ago (Cachin, 1998; Hopper, 2004). In 1998, Cachin (Cachin, 1998) modeled steganography systems from an information-theoretic perspective, using Kullback–Leibler (KL) divergence between cover and stego distributions

to evaluate security, with perfect security achieved when the KL divergence is zero. From a computational complexity perspective, Hooper (Hopper, 2004) laid the groundwork for secure steganography and provided example constructions. A stegosystem is considered secure if any probabilistic polynomial-time distinguisher cannot differentiate between the distributions of stego and cover, meaning they are computationally indistinguishable, thus making the stegosystem provably secure.

Recent advancements in generative models have enabled the simulation and generation of realistic carrier distributions, driving the practical application of provably secure steganography. For textual carriers, the rise of Large Language Models has made text distributions more accessible. State-of-the-art methods like METEOR (Kaptchuk et al., 2021), DISCOP (Ding et al., 2023), and iMEC (de Witt et al., 2022), with formal security proofs, embed secret bits by manipulating the sampling process based on the predicted symbol distributions. A detailed introduction to provably secure steganography can be found in the Appendix A.

However, the quality of the stego generated by the aforementioned provably secure methods depends entirely on the explicit distributions provided by generative models. In real-world scenarios, many model owners who offer high-quality generative models do not disclose the underlying distributions, as they seek to protect their intellectual property and prevent targeted attacks such as model distillation. Instead, they only provide the final generated output. This lack of access to model distributions makes it difficult to apply these provably secure methods in black-box scenarios where the model’s internal probability is unknown.

Unlike the aforementioned provably secure steganography algorithms that rely on white-box models, we propose a provably secure steganography algorithm effective in black-box scenarios. Our method relies on sampling rather than explicit distributions to embed secret information. The sender and receiver communicate securely through the sampling results transmitted by the channel.

Our main contributions are as follows:

1. We propose a provably secure steganography scheme based on sampling and provide formal security proof.
2. We propose an adaptive dynamic steganographic sampling method that adjusts the number of samples based on the current sampling collision

situation, aiming to balance collision and embedding capacity.

3. Extensive experiments show that our scheme is both effective and efficient. In black-box scenarios, it achieves comparable or even superior embedding capacity and high efficiency to white-box methods, while ensuring guaranteed security.

2 Proposed Method

2.1 Definition

Formally, our steganography scheme $\sum_{\mathcal{C}}$ is a triple of probabilistic algorithms $\sum_{\mathcal{C}} = (\mathbf{KeyGen}_{\mathcal{C}}, \mathbf{Enc}_{\mathcal{C}}, \mathbf{Dec}_{\mathcal{C}})$ parameterized by a sampleable covert channel \mathcal{C} .

- **KeyGen $_{\mathcal{C}}(1^\lambda)$** takes arbitrary input with length λ and generates the key \mathbf{K} shared between the sender and receiver.
- **Enc $_{\mathcal{C}}(\mathbf{K}, \mathbf{m}, \mathbf{H})$** takes the key K , a secret message $\mathbf{m} \in \{0, 1\}^*$ and the channel history $\mathbf{H} = \{h_0, h_1, \dots, h_{|\mathbf{H}|-1}\}$ and returns a stegotext $\mathbf{s} = s_1 || s_2 || \dots || s_n$, which is a symbol sequence with $s_i \in \mathcal{C}$.
- **Dec $_{\mathcal{C}}(\mathbf{K}, \mathbf{s}, \mathbf{H})$** takes as input the key K , the stegotext \mathbf{s} and the history \mathbf{H} and returns the extracted secret message \mathbf{m} on success or an empty string on failure.

The steganography scheme $\sum_{\mathcal{C}}$ should satisfy *correctness* and *security*.

- **Correctness:** $\sum_{\mathcal{C}}$ is *correct* if for any history \mathbf{H} and any message \mathbf{m} , an encoded secret message should be recover by the decoding algorithm only with negligible probability of error.

$$Pr[\mathbf{Dec}_{\mathcal{C}}(\mathbf{K}, \mathbf{Enc}_{\mathcal{C}}(\mathbf{K}, \mathbf{m}, \mathbf{H}), \mathbf{H}) = \mathbf{m}] \geq 1 - \text{negl}(\lambda), \quad (1)$$

where $\text{negl}(\lambda)$ is the negligible function that correlates to the secure parameter λ .

- **Security:** $\sum_{\mathcal{C}}$ is *computationally indistinguishable* against *chosen hiddentext attack* if for all polynomial-time adversaries $\mathcal{A}_{\mathcal{C}}$, for all \mathbf{K} :

$$|Pr[\mathcal{A}_{\mathcal{C}}^{\mathbf{Enc}_{\mathcal{C}}(\mathbf{K}, \cdot, \cdot)} = 1] - Pr[\mathcal{A}_{\mathcal{C}}^{\mathcal{O}_{\mathcal{C}}(\cdot, \cdot)} = 1]| \leq \text{negl}(\lambda), \quad (2)$$

where $\mathcal{O}_{\mathcal{C}}(\cdot, \cdot)$ is an oracle that can sample randomly from cover distribution \mathcal{C} .

2.2 Intuition

High-level sketch: At a high level, our steganography scheme operates autoregressively; each step, we sample multiple candidate from the *efficiently sampleable channel* using the secret key at that time. We map the secret information onto the samples taken at each time step. To illustrate with a simple example in Figure 1, if we sample four times (random number), the four sampling results can correspond to fragments of the secret information: "00", "01", "10", and "11" respectively. If the secret information corresponds uniquely to a sampling result, we can decode the secret information at that moment with a unique extraction result. However, if the sampling result corresponding to the secret information is the same as that of another possible piece of information, this is known as a *collision* (time $t + 2$ in Figure 1). A collision does not affect the embedding process of the secret information. In fact, it can still provide useful information to help us rule out some possible secret messages. We do not avoid situations where collisions occur. We will explain this in greater detail in later sections.

2.3 Steganography Scheme

2.3.1 Encoding

Our embedding method can be divided into two steps: determining the number of samples, sampling and identifying conflict intervals. The pseudocode for the overall encoding algorithm is presented in Algorithm 1, while the pseudocode for the adaptive dynamic sampling algorithm at each time step is shown in Algorithm 2.

Determining the number of samples At each time step, we do not sample deterministically; instead, we dynamically adjust the number of samples based on the conflicts from the previous step. The intuition is that if many conflicts occurred previously, we reduce the number of samples this time to avoid conflicts, implicitly increasing the unique decoding probability.

Specifically, we record the size of the collision set CS at each time step to determine the number of samples for the next step. Unlike instant code construction, we fully utilize the accumulated information from each time step. At every step, we expand the secret information to be embedded, with the number of expansions determined by $\log_2(2^N/|CS|)$, where N is the maximum number of secret bits that can be expanded at once.

This function is inspired by TCP flow control with sliding window adjustments. If a collision occurs, the window size decreases; otherwise, it increases multiplicatively. Once the expansion length L is determined, we expand the secret information by adding a bit (0 or 1). In decimal, this corresponds to multiplying by 2 or by 2 and adding 1. The resulting $expand_list$ contains the bits to be embedded, and its size determines the number of samples for this step.

Sampling and Identifying Collision Set

In the sampling phase, the only difference from standard language model sampling is that we sample $|expand_list|$ samples at once with replacement. We first determine the token corresponding to the secret information as the output s^* for this time step in the language model. At the same time, we identify all other $expand_list$ values that fall within the result of s^* , building a new conflict list CS (which must include at least one item—the secret information itself). Finally, the secret information’s bit index is shifted by L positions, and the updated conflict list is carried into the next autoregressive time step.

Moreover, since the source distribution for multiple sampling attempts at each step remains the same, we can leverage fast sampling methods, such as alias sampling (Walker, 1977, 1974), to efficiently draw multiple samples from the same distribution. This ensures that the sampling process does not impose a significant time burden on the overall steganographic procedure.

Algorithm 1 Main Loop of Secret Message Encoding

Input: Language Model M , History H , Shared Key \mathbf{K} , Secret Message \mathbf{m} , Max Embedded Bits N .

Output: Stego Message S .

- 1: $S \leftarrow \emptyset$
 - 2: $CS^0 \leftarrow \{0, 1\}$
 - 3: **while** not the end of m **And** $EOS \notin S$ **do**
 - 4: $C^t \leftarrow M(H)$
 - 5: $s^*, CS^t, bits^t \leftarrow ADS(C^t, \mathbf{m}, \mathbf{K}, CS^{t-1}, N, bits^{t-1})$
 - 6: $H \leftarrow H || s^*$
 - 7: $S \leftarrow S || s^*$
 - 8: **end while**
 - 9: **return** S
-

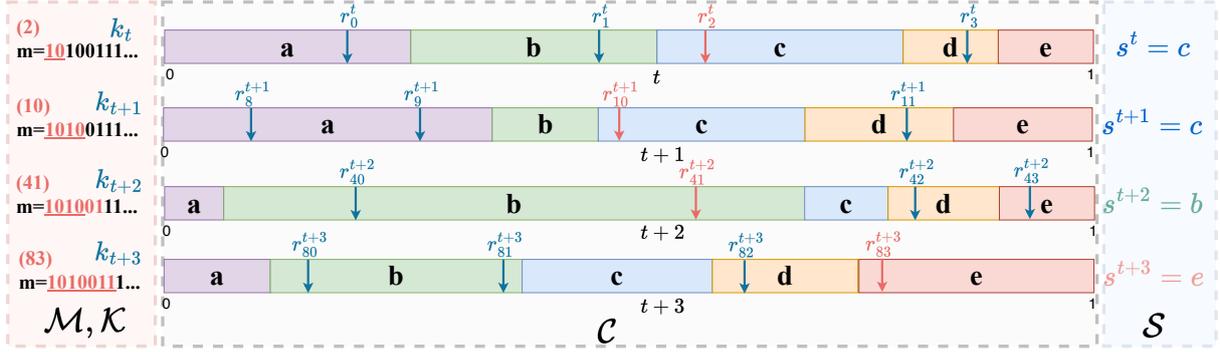


Figure 1: An example of our method when the maximum sampling attempts are set to 4 at each time step. Here, m represents the secret information, and s^t is the steganographic token output at time t . The red arrows \downarrow indicate the sampling results corresponding to the secret information, red secret information denotes the already embedded secret bits, and underlined text represents the secret information that can be immediately extracted.

2.3.2 Decoding

The decoding process mirrors the encoding process, with the algorithm executing the same two stages, as shown in Algorithm 3. To be specific, the receiver inputs the stego text S into the generative model M autoregressively to obtain a sampleable channel. Similar to the sending process, the receiver calculates the required number of samples at the current timestep t and samples from the channel. By observing how many sampling results match the current stego token s^* , an updated collision set CS' is constructed. If the collision set contains only one element, it means the sampling result representing the secret information is unique, allowing deterministic decoding of all secret message m up to the current timestep. If the collision set contains more than one element, the new collision set is recorded for decoding in the next time step. At this point, the determinable secret information m corresponds to the common prefix of the binary representations of all elements in the collision set. This process continues until all stego tokens are processed, yielding the fully recovered secret message m . The specific decoding algorithm for a single time step is shown in Algorithm 4.

2.4 Proof of security

Firstly we consider the case where only 1 bit of secret message ($m_i = 0$ or 1) is embedded. $P[m_i = 0] + P[m_i = 1] = 1$. Then we sample twice from the model distribution can get 2 sample results, namely c_0 and c_1 . Considering any symbol a from the model alphabet, we assume $P_S[a]$ is the probability of token a being selected within our steganography framework which can be calculated

as:

$$\begin{aligned}
 P_S[a] &= P[m_i = 0] \cdot P_S[a|m_i = 0] + P[m_i = 1] \cdot P_S[a|m_i = 1] \\
 &= P[m_i = 0] \cdot P[c_0 = a] + P[m_i = 1] \cdot P[c_1 = a] \\
 &= P[m_i = 0] \cdot P_M[a] + P[m_i = 1] \cdot P_M[a] \\
 &= P_M[a]
 \end{aligned}$$

Similarly, this conclusion also holds when embedding multiple bits of information because our steganography schemes only sample multiple times from the model and do not distort the original model distribution. A more rigorous hybrid proof of the security of our method is provided in the appendix B.

3 Experiments and Analysis

3.1 Experiment Settings

Task Our steganography method is essentially a universal steganography method that, in theory, can be compatible with any sequential form of carriers, as long as sampling results from the carrier distribution are available. It is well-suited for black-box scenarios where the detailed distribution is unknown. In the following discussion, We primarily use text generation tasks to demonstrate out scheme's effectiveness, as text generation with large language models is currently widely popular.

Baseline We evaluated our adaptive dynamic sampling steganography method using the random sampling decoding approach, alongside the ideal steganography algorithm and provably secure generative steganography algorithms with 100% deterministic decoding: METEOR (Kaptchuk et al., 2021) and DISCOP (Ding et al., 2023), as well

Algorithm 2 ADAPTIVEDYNAMICSAMPLING (ADS)

Input: Sampleable Channel C , Secret Message m , Shared Key \mathbf{K} , Collision Set CS , Max Embedded Bits N , Bit Index $bits$.

Output: Selected Symbol s^* , Updated Collision Set CS' Updated Bit Index $bits$.

```
1:  $L \leftarrow \lfloor \log_2(2^N / |CS|) \rfloor$ 
2:  $CS' \leftarrow \emptyset$ 
3:  $expand\_list \leftarrow CS$ 
4: for  $j = 0$  to  $L - 1$  do
5:    $tmp \leftarrow expand\_list.copy()$ 
6:    $expand\_list \leftarrow \emptyset$ 
7:   for  $i \in tmp$  do
8:      $expand\_list.append(2 \cdot i)$ 
9:      $expand\_list.append(2 \cdot i + 1)$ 
10:  end for
11: end for
12:  $n \leftarrow |expand\_list|$ 
13:  $\{s_0, \dots, s_n\} \xleftarrow{\text{Sample with } \mathbf{K}} C$ 
14:  $index \xleftarrow{\text{map}} m[: bits + L]$ 
15:  $s^* \leftarrow s_{index}$ 
16: for  $x_j \in expand\_list$  do
17:   if  $s_j = s^*$  then
18:      $CS'.append(x_j)$ 
19:   end if
20: end for
21:  $bits+ = L$ 
22: return Selected Symbol  $s^*$ , Updated Collision Set  $CS'$ , Bit Index  $bits$ .
```

as its variant, DISCOP Reorder (DISCOP with capacity boost). DISCOP and its variant were implemented in Cython, while the others were implemented in Python.

Dataset We used three types of datasets to simulate cover text channels. The first is a uniform distribution, simulating high entropy situations where each token in the language model’s tokenizer vocabulary has an equal probability. The second is a QA dataset, using the WILD (Ni et al., 2023), a typical application scenario where the language model answers questions. Finally, we used the XHS (Bai et al., 2024) dataset, which simulates social media text in Chinese, characterized by an informal, fragmented style typical of social network communication.

Model We evaluate our steganography method on three large language models: MISTRAL3 (Jiang et al., 2023), QWEN2 (Bai et al., 2023), and

Algorithm 3 Main Loop of Secret Message Decoding

Input: Stego Message \mathbf{S} , Language Model M , History H , Shared Key \mathbf{K} , Max Embedded Bits N .

Output: Secret Message m .

```
1:  $m \leftarrow \emptyset$ 
2:  $CS^0 \leftarrow \{0, 1\}$ 
3: Parse  $\mathbf{S} = \{s_0, s_1, \dots, s_{|c|-1}\}$ 
4: for  $i \in 0, 1, \dots, |c| - 1$  do
5:    $C^t \leftarrow M(H || \mathbf{S}[i])$ 
6:    $m, CS^t, bits^t \xleftarrow{\text{ADS}^{-1}(C^t, s_i, \mathbf{K}, CS^{t-1}, N, bits^{t-1})}$ 
7: end for
8: return  $m$ 
```

LLAMA3 (Dubey et al., 2024), using the WILD and XHS datasets, each with a model scale of 7B parameters. We customize the vocabulary size for testing under the uniform distribution. The prompts used for WILD and XHS datasets correspond to the questions within each dataset. These prompts are loaded into the function `apply_chat_template` and then become the input of LLMs. For each test, we let the models generate 1,000 samples with $top-p = 1.0$, $base\ temperature = 1.0$, the max generated length is 500. All our experiments were conducted on $2 \times$ NVIDIA A5000 GPUs (32GB RAM) and $24 \times$ Intel Xeon w5-3423 CPUs.

3.2 Metrics

We evaluate the performance of our steganography scheme in terms of **Security, Efficiency, and Capacity**.

3.2.1 Security

Although our steganography scheme’s computational security has been theoretically proven in the Methods section, we further validate its security through linguistic steganalysis in adversarial scenarios and assess its intuitive imperceptibility based on linguistic quality.

Steganalysis. We utilized three classic steganalysis models to assess the imperceptibility of stegotexts: TS-CSW (Yang et al., 2020b) based on pre-trained BERT (Liu et al., 2023). We trained the detectors on 1,000 samples of generated stegotext and corresponding cover texts, randomly sampled from the same language model, using a 3:1:1 split for training, validation, and testing. With a learning rate of $1e-4$, training was conducted for three

Algorithm 4 ADAPTIVEDYNAMICSAMPLING⁻¹
(ADS⁻¹)

Input: Sampleable Channel C , Shared Key \mathbf{K} , Collision Set CS , Max Embedded Bits N , Bit Index bit , Selected Symbol s^* .

Output: Decoded Secret Message \mathbf{m} , Updated Collision Set CS' .

```
1:  $L \leftarrow \lfloor \log_2(2^N/|CS|) \rfloor$ 
2:  $CS' \leftarrow \emptyset$ 
3:  $expand\_list \leftarrow CS$ 
4: for  $j = 0$  to  $L - 1$  do
5:    $tmp \leftarrow expand\_list.copy()$ 
6:    $expand\_list \leftarrow \emptyset$ 
7:   for  $i \in tmp$  do
8:      $expand\_list.append(2 \cdot i)$ 
9:      $expand\_list.append(2 \cdot i + 1)$ 
10:  end for
11: end for
12:  $n \leftarrow |expand\_list|$ 
13:  $\{s_0, \dots, s_n\} \xleftarrow{\text{Sample with } \mathbf{K}} p$ 
14: for  $s_j \in \{s_0, \dots, s_n\}$  do
15:   if  $s_j = s^*$  then
16:      $CS'.append(expand\_list[j])$ 
17:   end if
18: end for
19: if  $|CS'| = 1$  then
20:    $\mathbf{m} \xleftarrow{map} CS'[0]$ 
21: end if
22:  $\mathbf{m} \xleftarrow{CommonPrefix-map} CS'$ 
23:  $bits+ = L$ 
24: Return: Decoded Secret Message  $\mathbf{m}$ , Updated Collision Set  $CS'$ , Bit Index  $bits$ .
```

epochs. After repeating the process three times, the average F1 score from the test set was used as the steganalysis F1 score.

Linguistic Quality. We evaluate linguistic quality using perplexity (PPL) and diversity. PPL measures the fluency of generated text, with lower PPL values indicating smoother and more natural text.

$$PPL = \exp \left(-\frac{1}{N} \sum_{i=1}^N \log p(x_i | \mathbf{x}_{1:i-1}) \right). \quad (3)$$

As for text diversity, we used the $distinct_n$ metric. This metric needs to find the unique pieces of tokens in the text and calculate their ratio.

$$distinct_n = \frac{\text{count}(\text{unique } n\text{-grams})}{\text{count}(n\text{-grams})}. \quad (4)$$

3.2.2 Efficiency

Time. For real-time covert communication, it's important to complete stego embedding quickly. To measure this efficiency, We measure the average time of stego generation, calculated by dividing the total time taken for the entire process by the length of the embedded message. The shorter, the better.

3.2.3 Capacity

Entropy. This represents the theoretical upper bound of the embedding capacity, measured in bits per symbol. For texts, a symbol usually refers to a token generated by large language models (LLMs). Entropy is calculated as: $Entropy = \frac{1}{N} \sum_i^N p(x_i | x_{<i}) \log_2 p(x_i | x_{<i})$.

Embedding Capacity. This metric measures the average number of bits that can be successfully embedded and extracted, expressed in bits per token.

Utilization. This metric reflects the efficiency of the embedding process, representing the ratio of the actual embedding capacity to the theoretical capacity limit (Entropy). The larger the number, the better.

3.3 Evaluation Results and Discussion

3.3.1 On the Capacity & Entropy

Although we do not use explicit distribution information, we prove that with a sufficiently large number of samples, our method can still approximate entropy, achieving a steganographic capacity comparable to, or even better than, methods in white-box scenarios. As shown in Tables 1 and 2, our method achieves the highest embedding capacity and entropy utilization, even outperforming DISCOP Reorder, which has the best capacity under white-box distribution. Suppose at a certain time we sample samples from a uniform distribution of size n , with one sample representing the secret information. The size of the set of all possible secret messages is m . Samples that do not conflict with the one representing the secret information can be eliminated during decoding, as their corresponding secret information can be ruled out. The expected number of secret information that can

Model	Algorithm	Capacity			Time	Linguistic Quality				Imperceptibility
		Entropy/token	Embed bit/token \uparrow	Utili. \uparrow	Time \downarrow	PPL	Dist2	Dist3	Dist4	TS-CSW F1 \downarrow
MISTRAL	Random Sampling	0.9489	0	0	0.0295	1.9890	0.6073	0.8377	0.9208	-
	METEOR	0.9870	0.5732	0.5633	0.0298	2.0888	0.6135	0.8458	0.9275	54.23% $_{3.95}$
	DISCOP (R.)	0.9506	0.7683	0.7930	0.0427	2.0141	0.6086	0.8389	0.9215	51.42% $_{1.24}$
	DISCOP	0.9267	0.4242	0.4631	0.0379	1.9848	0.6044	0.8351	0.9192	51.51% $_{1.32}$
	Ours	0.9872	0.8964	0.9137	0.0295	2.0788	0.6097	0.8415	0.9246	50.18% $_{1.62}$
QWEN	RandomSampling	0.6934	0	0	0.0308	2.7253	0.5984	0.8298	0.9153	-
	METEOR	0.7172	0.3977	0.5502	0.0312	2.7979	0.5989	0.8349	0.9203	54.75% $_{2.26}$
	DISCOP(R.)	0.6955	0.5166	0.7336	0.0756	2.7103	0.5958	0.8303	0.9147	50.84% $_{2.01}$
	DISCOP	0.6928	0.3261	0.4741	0.0533	2.76938	0.5974	0.8356	0.9238	49.94% $_{1.39}$
	Ours	0.7068	0.6491	0.9251	0.0308	2.7567	0.6024	0.8372	0.9204	49.83% $_{1.58}$
LLAMA3	RandomSampling	0.6934	0	0	0.0312	1.7340	0.6090	0.8399	0.9222	-
	METEOR	0.7172	0.4031	0.5574	0.0317	1.7815	0.6100	0.8465	0.9284	53.86% $_{2.39}$
	DISCOP(R.)	0.6955	0.5221	0.7425	0.0765	1.7083	0.6060	0.8408	0.9219	50.28% $_{0.96}$
	DISCOP	0.6928	0.3297	0.4795	0.0540	1.7405	0.6078	0.8456	0.9301	49.98% $_{1.57}$
	Ours	0.6983	0.5861	0.8393	0.0313	1.6896	0.6063	0.8339	0.9260	49.73% $_{2.79}$

Table 1: Main Results on WILD. Higher capacity and utilization are better. Lower time is better. Linguistic quality closer to random sampling is better. In our method, the number of sampling iterations is fixed at 2^9 .

Model	Algorithm	Capacity			Time	Linguistic Quality				Imperceptibility
		Entropy/token	Embed bit/token	Utili.	Time	PPL	Dist2	Dist3	Dist4	TS-CSW F1 \downarrow
MISTRAL	RandomSampling	2.1963	0	0	0.0294	4.8066	0.5367	0.7639	0.8657	-
	METEOR	2.2195	1.5028	0.6635	0.0299	4.8673	0.5183	0.7560	0.8614	52.91% $_{3.83}$
	DISCOP(R.)	2.2863	2.0416	0.8825	0.0427	5.0154	0.5345	0.7723	0.8792	51.84% $_{3.67}$
	DISCOP	2.2291	0.8450	0.3837	0.0370	4.8425	0.5219	0.7564	0.8620	50.97% $_{3.03}$
	Ours	2.1986	2.0378	0.9308	0.0298	4.9094	0.5341	0.7675	0.8721	50.81% $_{2.12}$
QWEN	RandomSampling	3.5673	0	0	0.0319	12.6965	0.7527	0.9198	0.9585	-
	METEOR	3.6247	2.5283	0.6892	0.0328	13.4175	0.7634	0.9309	0.9704	52.38% $_{3.10}$
	DISCOP(R.)	3.5478	3.2606	0.9118	0.0824	14.7447	0.7498	0.9171	0.9542	50.28% $_{2.64}$
	DISCOP	3.5371	1.3595	0.3852	0.0523	12.8203	0.7365	0.9066	0.9467	50.03% $_{2.96}$
	Ours	3.7272	3.4928	0.9393	0.0299	13.3314	0.7642	0.9209	0.9588	49.83% $_{2.58}$
LLAMA3	plain	1.3315	0	0	0.0313	3.3848	0.5394	0.7649	0.8635	-
	METEOR	1.3410	0.8466	0.5760	0.0319	7.4924	0.5427	0.7720	0.8743	52.38% $_{3.10}$
	DISCOP(R.)	1.4737	1.2137	0.8243	0.0762	4.9633	0.5583	0.7762	0.8721	50.01% $_{2.70}$
	DISCOP	1.3739	0.5625	0.4380	0.0519	4.3652	0.5399	0.7639	0.8678	49.98% $_{2.09}$
	Ours	1.4267	1.2560	0.9223	0.0314	4.3966	0.5447	0.7727	0.8745	49.66% $_{2.58}$

Table 2: Main Results on XHS. Higher capacity and utilization are better. Lower time is better. Linguistic quality closer to random sampling is better. In our method, the number of sampling iterations is fixed at 2^9 .

be eliminated $E(e)$ at this time step is:

$$\begin{aligned}
E(e) &= \sum_{k=0}^{m-1} (m-1-k) \binom{m-1}{k} \frac{(n-1)^{m-1-k}}{n^{m-1}} \\
&= (m-1)! \frac{(n-1)^{m-1}}{n^{m-1}} \sum_{k=0}^{m-1} \frac{(n-1)^{-k}}{k!(m-2-k)!} \\
&= (m-1)! \frac{(n-1)^{m-1}}{n^{m-1}} \frac{(n-1)^{m+1} n^m}{n^{m+1} (n-1)^m} \\
&= (m-1) \frac{n-1}{n}.
\end{aligned}$$

When $m \rightarrow n$, the expectation approaches $m-1$, meaning that in each step, the secret message can be uniquely decoded and recovered.

The results from the uniform distribution show that as the maximum allowed sampling approaches

the vocabulary size, the embedding capacity asymptotically approaches the entropy of the uniform distribution. More sampling iterations result in higher embedding capacity. This trend holds across all three models with different vocabulary sizes. For all models and scenarios, our method achieves high entropy utilization, even when the distribution is unknown. However, the upper limit of embedding capacity still depends on the distribution’s entropy.

3.3.2 Time Efficiency

The runtime of our method is primarily constrained by two factors: sampling time and the generation speed of the generative model. Although increasing the number of samples does raise the time required, it does not impose a significant burden and

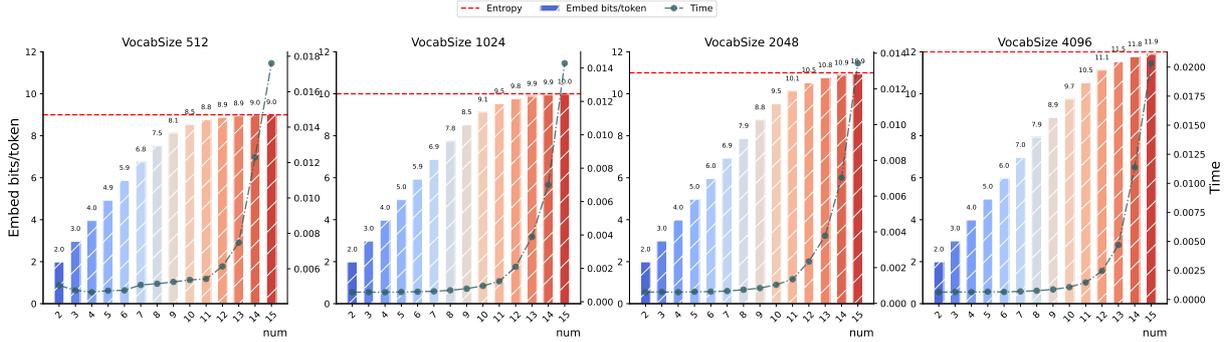


Figure 2: The relationship between the number of uniform distribution samples and embedding capacity. num represents the number of samples as 2^{num} , the red dashed line indicates entropy, and the green line represents the generation time per token.

remains within an acceptable range. On average, our method takes less than 100 milliseconds to process each token. On the WILD QA dataset, the Table 1 shows that our method is on the same time scale as existing provably secure steganography algorithms and can be faster than the reordering distribution version of DISCOP. On the social network XHS dataset in Table 2, our method’s runtime is nearly identical to that of standard random sampling algorithms, and its time performance is comparable to existing provably secure algorithms such as METEOR and the non-reordering version of DISCOP, demonstrating the time efficiency of our approach. In the uniform distribution scenario, when the number of samples does not exceed 2^{14} , our method achieves the same generation speed as random sampling. This trend is consistently observed across the LLAMA3, MISTRAL, and QWEN models. Our implementation is in Python, so performance could be significantly optimized by replacing the sampling component with C/C++. This would bring the encoding speed closer to the model’s generation rate.

3.3.3 On the Generation Quality of Stego

In Tables 1 and 2, we present the experimental results on the XHS dataset and the QA dataset. (In the uniform distribution scenario, there is no need to discuss the quality of the generated text.) It is evident that, in the text generation task, the stegotext produced by our method exhibits linguistic quality similar to that of text generated through random sampling, both in terms of PPL (perplexity) and diversity metrics. This aligns with our security proof, demonstrating that the stegotext generated by our method is indistinguishable from the cover text. In terms of fluency, our method achieved PPL scores similar to those of texts generated through random

sampling. Higher PPL scores can sometimes result from increased entropy. In fact, studies have shown that blindly aiming for low PPL scores can lead to bland, redundant text or repetitive loops (Holtzman et al., 2019). However, the texts generated by our method did not exhibit these undesirable patterns and maintained good readability.

4 Conclusion

In this paper, we propose a provably secure steganography scheme for black-box scenarios based on Adaptive Dynamic Sampling. Unlike existing methods that rely on explicit model distributions to conceal secret messages, our approach only requires random sampling results. We design a strategy to determine the number of samples at each step and embed secret messages into the sampling indices without altering the model’s original generation process, ensuring theoretical indistinguishability between stego and cover. Extensive evaluations on three popular LLMs and three benchmark datasets show that our scheme achieves high generative quality, efficiency, and capacity, comparable to white-box provably secure methods.

5 Ethics Statement

We propose a steganography framework based on LLMs. Given the ease of accessing LLMs, this method may pose a risk of abuse with texts generated by these models. In future work, we will explore detection methods for LLM-generated steganographic texts. In our implementation and experiments, we adhere to the licenses of the scientific artifacts used.

References

- Jinze Bai, Shuai Bai, and Yunfei Chu et. al. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.
- Yuelin Bai, Xinrun Du, Yiming Liang, Yonggang Jin, Ziqiang Liu, Junting Zhou, Tianyu Zheng, Xincheng Zhang, Nuo Ma, Zekun Wang, et al. 2024. Coig-cqia: Quality is all you need for chinese instruction fine-tuning.
- R.J. Barron, Brian Chen, and G.W. Wornell. 2003. The duality between information embedding and source coding with side information and some applications. *IEEE Transactions on Information Theory*, 49(5):1159–1180.
- Christian Cachin. 1998. An information-theoretic model for steganography. In *International Workshop on Information Hiding*, pages 306–318. Springer.
- Ferdinando Cicalese, Luisa Gargano, and Ugo Vaccaro. 2019. Minimum-entropy couplings and their applications. *IEEE Transactions on Information Theory*, 65(6):3436–3451.
- Rémi Cogramne, Quentin Giboulot, and Patrick Bas. 2022. Efficient steganography in jpeg images by minimizing performance of optimal detector. *IEEE Transactions on Information Forensics and Security*, 17:1328–1343.
- Christian Schroeder de Witt, Samuel Sokota, J Zico Kolter, Jakob Nicolaus Foerster, and Martin Strohmaier. 2022. Perfectly secure steganography using minimum entropy coupling. In *The Eleventh International Conference on Learning Representations*.
- Jinyang Ding, Kejiang Chen, Yaofei Wang, Na Zhao, Weiming Zhang, and Nenghai Yu. 2023. Discop: Provably secure steganography in practice based on “distribution copies”. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 2238–2255. IEEE Computer Society.
- Roger Dingledine, Nick Mathewson, Paul F Syverson, et al. 2004. Tor: The second-generation onion router. In *USENIX security symposium*, volume 4, pages 303–320.
- Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.
- Tina Fang, Martin Jaggi, and Katerina Argyraki. 2017. Generating steganographic text with lstms. *ACL 2017*, page 100.
- Jessica Fridrich and Miroslav Goljan. 2004. On estimation of secret message length in lsb steganography in spatial domain. In *Security, steganography, and watermarking of multimedia contents VI*, volume 5306, pages 23–34. SPIE.
- Phillipa Gill, Masashi Crete-Nishihata, Jakub Dalek, Sharon Goldberg, Adam Senft, and Greg Wiseman. 2015. Characterizing web censorship worldwide: Another look at the opennet initiative data. *ACM Transactions on the Web (TWEB)*, 9(1):1–29.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *International Conference on Learning Representations*.
- Nicholas J Hopper. 2004. *Toward a theory of Steganography*. Carnegie Mellon University.
- Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.
- Gabriel Kaptchuk, Tushar M. Jois, Matthew Green, and Aviel D. Rubin. 2021. Meteor: Cryptographically secure steganography for realistic distributions. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, CCS ’21*, page 1529–1548, New York, NY, USA. Association for Computing Machinery.
- Murat Kocaoglu, Alexandros Dimakis, Sriram Vishwanath, and Babak Hassibi. 2017a. Entropic causal inference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.
- Murat Kocaoglu, Alexandros G. Dimakis, Sriram Vishwanath, and Babak Hassibi. 2017b. Entropic causality and greedy minimum entropy coupling. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 1465–1469.
- Bin Li, Ming Wang, Xiaolong Li, Shunquan Tan, and Jiwu Huang. 2015. A strategy of clustering modification directions in spatial image steganography. *IEEE Transactions on Information Forensics and Security*, 10(9):1905–1917.
- Cheuk Ting Li. 2021. Efficient approximate minimum entropy coupling of multiple probability distributions. *IEEE Transactions on Information Theory*, 67(8):5259–5268.
- Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Comput. Surv.*, 55(9).
- Qiang Liu, Xuyu Xiang, Jiaohua Qin, Yun Tan, Junshan Tan, and Yuanjing Luo. 2020. Coverless steganography based on image retrieval of densenet features and dwt sequence mapping. *Knowledge-Based Systems*, 192:105375.
- Jinjie Ni, Fuzhao Xue, Yuntian Deng, Jason Phang, Kabir Jain, Mahir Hitesh Shah, Zangwei Zheng, and Yang You. 2023. Instruction in the wild: A user-based instruction dataset. <https://github.com/XueFuzhao/InstructionWild>.
- Michael G Reed, Paul F Syverson, and David M Goldschlag. 1998. Anonymous connections and onion routing. *IEEE Journal on Selected areas in Communications*, 16(4):482–494.
- Esra Satir and Hakan Isik. 2014. A huffman compression based text steganography method. *Multimedia tools and applications*, 70:2085–2110.
- Mehdi Sharifzadeh, Mohammed Aloraini, and Dan Schonfeld. 2020. Adaptive batch size image merging steganography and quantized gaussian image steganography. *IEEE Transactions on Information Forensics and Security*, 15:867–879.
- Ram Sundara Raman, Prerana Shenoy, Katharina Kohls, and Roya Ensafi. 2020. Censored planet: An internet-wide, longitudinal censorship observatory. In *proceedings of the 2020 ACM SIGSAC conference on computer and communications security*, pages 49–66.
- Tri Van Le. 2003. Efficient provably secure public key steganography. *Cryptology ePrint Archive*.

Alastair J Walker. 1974. New fast method for generating discrete random numbers with arbitrary frequency distributions. *Electronics Letters*, 10(8):127–128.

Alastair J Walker. 1977. An efficient method for generating discrete random variables with general distributions. *ACM Transactions on Mathematical Software (TOMS)*, 3(3):253–256.

Yiming Xue, Jiaxuan Wu, Ronghua Ji, Ping Zhong, Juan Wen, and Wanli Peng. 2023. Adaptive domain-invariant feature extraction for cross-domain linguistic steganalysis. *IEEE Transactions on Information Forensics and Security*.

Zhong-Liang Yang, Xiao-Qing Guo, Zi-Ming Chen, Yong-Feng Huang, and Yu-Jin Zhang. 2018. Rnn-stega: Linguistic steganography based on recurrent neural networks. *IEEE Transactions on Information Forensics and Security*, 14(5):1280–1295.

Zhongliang Yang, Yongfeng Huang, and Yu-Jin Zhang. 2020a. Ts-csw: text steganalysis and hidden capacity estimation based on convolutional sliding windows. *Multimedia Tools and Applications*, 79(25):18293–18316.

Zhongliang Yang, Yongfeng Huang, and Yujin Zhang. 2020b. Ts-csw: Text steganalysis and hidden capacity estimation based on convolutional sliding windows. *Multimedia Tools and Applications*, 79:18293–18316.

Siyu Zhang, Zhongliang Yang, Jinshuai Yang, and Yongfeng Huang. 2021. Provably secure generative linguistic steganography. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3046–3055.

Zhili Zhou, Yan Mu, and QM Jonathan Wu. 2019. Coverless image steganography using partial-duplicate image retrieval. *Soft Computing*, 23(13):4927–4938.

Zachary Ziegler, Yuntian Deng, and Alexander M Rush. 2019. Neural linguistic steganography. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1210–1215.

A Related Works

A.1 Provable Secure Steganography

A.1.1 Definition of Security

There are two definitions of security in steganography. Cachin (Cachin, 1998) first modeled steganographic security by the Kullback-Leibler Divergence (KL-Divergence) between the stegotext and covertext approaching zero:

$$D_{KL}(P_S||P_C) = \sum_x P_S(x) \log \frac{P_S(x)}{P_C(x)} < \epsilon, \quad (5)$$

where $D_{KL}(P_S||P_C) = 0$ denotes perfect security. Different from the Cachin, Another widely used steganography security is based on the complexity theory in the perspective of adversarial *chosen hiddentext attacks*. They (Hopper, 2004) assume for all polynomial-time (PPT) adversaries \mathcal{A} ,

$k \leftarrow \text{KeyGen}(1^\lambda)$, the advantage of \mathcal{A} in the games that distinguishes stego and cover is negligible:

$$|\Pr[\mathcal{A}^{\mathcal{O}^{\text{Encode}}(k, \cdot, \cdot)} = 1] - \Pr[\mathcal{A}^{\mathcal{O}^R(\cdot, \cdot)} = 1]| < \text{negl}(\lambda), \quad (6)$$

where \mathcal{O}^R denotes random sampling, $\mathcal{O}^{\text{Encode}}$ represents the steganography sampling algorithm, and $\text{negl}(\lambda)$ is a negligible function.

A.1.2 Entropy Coding Based Steganography

Baron et al. (Barron et al., 2003) first introduced the duality between information embedding and source coding. Van Le (Van Le, 2003) explored the use of arithmetic coding (AC) in steganography, and Satir et al. (Satir and Isik, 2014) applied Huffman coding (HC) for the same purpose. These methods leverage entropy coding to assign codewords to symbols based on their probabilities. As the codewords are prefix-free, secret bits can be matched to a single codeword, and the corresponding symbol transmitted. However, these approaches suffer from mismatched probabilities between codewords and symbols, as symbol probabilities are not always negative integral powers of 2. This causes the output distribution to deviate from the channel distribution, making the method more detectable. Additionally, when symbol alphabets are large, constructing the codebook becomes time-consuming. Despite these limitations, entropy coding represents the theoretical upper limit for channel entropy utilization. With improved security measures, the focus will shift towards maximizing the steganographic embedding rate, with entropy coding offering valuable insights for future research.

A.1.3 METEOR

Kaptchuk et al. (Kaptchuk et al., 2021) proposed METEOR, a secure steganography that re-encrypts the embedded bits step by step. This work points out that previous informal methods (Yang et al., 2018; Ziegler et al., 2019) have the problem of randomness reuse, which will cause the leakage of hidden information. METEOR collects several tokens until the entropy is enough for embedding, and re-encrypts the rest of bits to avoid information leakage.

This method is perfectly secure but the entropy is not fully used. The bits are re-encrypted after a loop of embedding which can be seen as a truncation of coding. The truncation of the embedding process results that entropy is wasted at the end

of each embedding loop. When the entropy of channel is extremely low, this situation may occur frequently and results in low capacity.

In order to further improve the expectation of embedding capacity, an reorder algorithm is proposed for METEOR. The complexity of the reorder algorithm is linearly related to the size of alphabet. Reordering the distributions increases capacity by 20%-25%.

A.1.4 Distribution Copies

Ding et al.(Ding et al., 2023) proposed a provably secure steganography method based on the copies of distribution. In the situation that the random numbers $r \in [0, 1)$ and $r + \frac{1}{2} \bmod 1$ point to different tokens in the copies, these 2 special tokens can be chosen according to the secret bit 0 or 1. If the random numbers $r, r + \frac{1}{2^n} \bmod 1, r + \frac{2}{2^n} \bmod 1, \dots, r + \frac{2^n-1}{2^n} \bmod 1$ point to different tokens, the embedding capacity is n bits.

Since copies of distribution may have lots of overlapping zones, the embedding rate is highly restricted. Ding et al.(Ding et al., 2023) proved that the embedding rate is asymptotic to the minimum entropy, computed in token level.

In order to further improve the embedding capacity, they proposed a recursion version of DISCOP. The distribution is reordered by a binary tree that keeps the probability of nodes as balance as possible, which is similar to a huffman tree. However, this process needs lots of time and thus makes it not efficient.

A.1.5 Minimum Entropy Coupling (MEC)

Minimum Entropy Coupling (MEC) is the problem that given the marginal distribution of the random variables, compute the joint distribution that has the minimum entropy. The core of this problem is to find how relevant it is likely to be between these variables. More details can be found in excellent work of Cicalese et al. (Cicalese et al., 2019).

De Witt et al.(de Witt et al., 2022) first explore using MEC to construct secure and efficient steganography. They try to compute MEC matrix between the uniform distributed n -bit codewords and the distribution predicted by language models. Then they choose the n -bit prefix of secret messages as the chosen codeword, and randomly sample a token from the marginal distribution of this codeword. Usually the n -bit codewords cannot be embedded in one loop, this process will be repeated several times until the bits can be uniquely

decoded.

However, computing the minimum entropy coupling is a NP-Hard problem. Many efforts have been made to design a poly-time algorithm to asymptotically approach the MEC (Kocaoglu et al., 2017a; Cicalese et al., 2019; Kocaoglu et al., 2017b; Li, 2021).

MEC-based steganography has 2 weaknesses: (i) up to now the best MEC algorithm is only able to compute a joint distribution whose entropy is within 1 bit of MEC, resulting in sustained loss of capacity and (ii) the complexity is at least $O(n \log n)$ (Cicalese et al., 2019). Therefore the large set of codewords will make this type of steganography not efficient. Beside these, there are still some minor issues like decoding error and dead cycles in practical. MEC-based steganography still needs more exploration.

B Proof of security

From a more formal standpoint, we prove the computational-indistinguishable security by a sequence of games $G_0 : G_1 : G_2 : G_3$, as shown in Figure 3. The security of our framework relies on the pseudo-random generator and the i.i.d. property of samples drawn from the same distribution. $G_0 = G_1$: G_0 is the steganographic sampling procedure controlled by the secret bit, while G_1 is independent of secret bit.

$$\begin{aligned} P_S[a] &= \sum_{m_i \in M} P[m_i] \cdot P_S[a|m_i] \\ &= \sum_{m_i \in M} P[m_i] \cdot P_M[a] \\ &= P_M[a] \end{aligned}$$

$G_1 \approx_c G_2$: G_2 uses random generator $\text{RANDOMORACLE}(\cdot)$ ($R(\cdot)$) to substitute the pseudo-random generator $\text{PRG}_k(\cdot)$ ($G_k(\cdot)$) in G_1 . From the definition of pseudo-random generator, for any probabilistic polynomial time (p.p.t.) adversary \mathcal{A} ,

$$\left| \mathbb{P} \left\{ \mathcal{A}^{\text{PRG}_k(\cdot)}(1^k) = 1 \right\} - \mathbb{P} \left\{ \mathcal{A}^{R(\cdot)}(1^k) = 1 \right\} \right| \leq \text{negl}(k). \quad (7)$$

Assuming the existence of a p.p.t. adversary \mathcal{A}' , which can distinguish the output of G_1 and G_2 with a noticeable probability, we can use the adversary \mathcal{A}' to construct another p.p.t. adversary \mathcal{A}'' to distinguish between the output of random generator $R(\cdot)$ and pseudo-random generator $G_k(\cdot)$. \mathcal{A}'' runs G_1 and G_2 with random generator $R(\cdot)$ and pseudo-random generator $G_k(\cdot)$. Though now \mathcal{A}'' can neither distinguish between $R(\cdot)$ and $G_k(\cdot)$ nor G_1 and G_2 , he can query \mathcal{A}' for each pair of

output of G_1 and G_2 . Since \mathcal{A}' can distinguish the output of G_1 and G_2 with a noticeable probability, the adversary \mathcal{A}'' can distinguish the output of random generator $R(\cdot)$ and pseudo-random generator $G_k(\cdot)$ with a noticeable probability, which contradicts the definition of pseudo-random generator. Therefore, such adversary \mathcal{A}' does not exist, and the distribution of output of G_1 and G_2 is computationally indistinguishable. In the case where $G_2 = G_3$: G_2 leverages the independent and identically distributed (i.i.d.) property of samples drawn from the same distribution. Since the output of the sampling process adheres to the model's distribution, randomly sampling a result from multiple samplings is equivalent to directly sampling a result from the model itself. Up to this point, we have demonstrated the security of our steganographic construction, demonstrating that it achieves computational indistinguishability from random sampling results unaffected by the steganographic embedding within the model.

C Demo of generated stegotexts

We present demo stegotexts generated by existing provably secure generative steganography methods and our own steganography constructions in Table 3.

G_0 : Steganography-Sampler

```

1:  $r \leftarrow \text{PRG}(k)$ 
2: for  $i \in 1, \dots, n$  do
3:   Sample token  $t_i \sim M$  with  $r$ 
4: end for
5: Index for  $t_1, t_2, \dots, t_n$ 
6: Map message  $m[1 : N] \rightarrow M \in \mathbb{N}^+$ 
7:  $M = \sum_{i=1}^N m_i \cdot 2^{i-1}$ 
8: return  $t_M$ 

```

G_1 : Pseudorandom Multiple Sampling

```

1:  $r \leftarrow \text{PRG}(k)$ 
2: for  $i \in 1, \dots, n$  do
3:   Sample token  $t_i \sim M$  with  $r$ 
4: end for
5: Index for  $t_1, t_2, \dots, t_n$ 
6: Randomly Sample a index  $M^*$ 
7: return  $t_{M^*}$ 

```

G_2 : Random Multiple Sampling

```

1: for  $i \in 1, \dots, n$  do
2:   Sample token  $t_i \sim M$  with RANDO-
      MORACLE
3: end for
4: Index for  $t_1, t_2, \dots, t_n$ 
5: Randomly Sample a index  $M^*$ 
6: return  $t_{M^*}$ 

```

G_3 : Model Sampling

```

1: Sample a token  $t \sim M$ 
2: return  $t$ 

```

Figure 3: Games used in the proof of steganography security.

Table 3: Examples in WILD with LLAMA3

Method	Generated Texts
Prompt	Please provide an action plan for reducing carbon footprint on a corporate level.
Random Sampling	<p>Here is a comprehensive action plan for reducing carbon footprint on a corporate level:</p> <p>**I. Set Goals and Track Progress**</p> <ol style="list-style-type: none"> 1. Set specific, measurable, achievable, relevant, and time-bound (SMART) targets for reducing carbon footprint. 2. Establish a carbon footprint tracking system to monitor progress and identify areas for improvement. 3. Report carbon footprint reduction milestones to stakeholders, including employees, customers, and investors...
METEOR (Kaptchuk et al., 2021)	<p>Here's a comprehensive action plan for reducing carbon footprint on a corporate level:</p> <p>**I. Assessment and Goal-Setting**</p> <ol style="list-style-type: none"> 1. Conduct a carbon footprint assessment to identify areas of impact: <ul style="list-style-type: none"> - Energy consumption (electricity, gas, water) - Transportation (company-owned vehicles, employee travel) - Supply chain emissions (procurement, logistics) - Waste management (recycling, disposal) 2. Set specific, measurable, achievable, relevant, and time-bound (SMART) targets...
DISCOP (Ding et al., 2023)	<p>Here is a comprehensive action plan for reducing carbon footprint on a corporate level:</p> <p>**I. Conduct a Carbon Footprint Assessment**</p> <ol style="list-style-type: none"> 1. Engage a third-party auditor or consultant to conduct a comprehensive carbon footprint assessment of your organization's operations, including Scope 1 (direct emissions), Scope 2 (indirect emissions), and Scope 3 (value chain emissions). 2. Identify key areas of energy consumption, such as: <ul style="list-style-type: none"> - Buildings and facilities - Transportation - Supply chain and...
DISCOP (R.)(Ding et al., 2023)	<p>Here's a comprehensive action plan for reducing a corporation's carbon footprint on a corporate level: **Step 1: Establish a Carbon Reduction Goal**</p> <ol style="list-style-type: none"> 1. Set a specific, measurable, achievable, relevant, and time-bound (SMART) goal to reduce carbon emissions. 2. Engage employees, stakeholders, and leadership in setting the goal and establishing a sense of ownership and accountability. 3. Conduct a carbon footprint assessment to identify areas for improvement and establish a baseline for measuring progress...
Ours	<p>Reducing a corporation's carbon footprint is a crucial step towards sustainability and environmental responsibility. Here is a comprehensive action plan for corporate carbon footprint reduction:</p> <p>**I. Conduct a Carbon Footprint Assessment**</p> <ol style="list-style-type: none"> 1. Identify the company's scope of activities: Determine what activities and products fall within the company's boundary for carbon footprint calculation. 2. Gather data: Collect data on energy consumption, water usage, waste production, and greenhouse gas emissions from various sources...